W numerze - blankiet zamówienia prenumeraty dla instytucji

P 1877/83

6 1983

informatyka

Programowanie współbieżne

TEXAS INSTRUMENTS 9900

Projektowanie systemu graficznego

Pamięci WINCHESTER

Prefiksowanie klasami

WYDAWNICTWO NOT SIGMA

Do autorów

Nie publikujemy precyzyjnych zaleceń dla autorów, by nie peszyć ich natłokiem dyrektyw i ograniczeń. Podstawowe zasady, jakimi kierujemy się przy kwalifikacji artykułu i warunki formalne, jakie powinien spełniać maszynopis, chcemy jednak co jakiś czas przypominać.

Wiele zasad oczywistych jest już przy pobieżnym przejrzeniu pisma: jak dzielić artykuł na części, co wyróżniać, w jakiej formie umieszczać bibliografię, jak zapisywać własny życiorys. Natomiast na elementarne pytanie — po co piszę — odpowiedzieć sobie musi przede wszystkim sam autor.

Podstawowym kryterium merytorycznym materiału (nie licząc głównego kryterium — oceny z punktu widzenia dotychczasowej wiedzy informatycznej) jest społeczna użyteczność zawartych w tekście rozważań i informacji. To w końcu oczywiste — jesteśmy powołani przede wszystkim do tego, aby służyć czytelnikom. W pierwszej kolejności liczyć się musi zatem interes społeczny. O tym winni także pamiętać autorzy.

Objętość tekstu powinna być wyznaczona przez społeczną wagę tematu, jego konstrukcja — przez wymóg czytelnego przekazu, zaś forma — przez dostosowanie języka i sposobu argumentacji do możliwości percepcyjnych przeciętnego odbiorcy. To truizmy, niemniej — jak praktyka wskazuje — warte przypomnienia. Autorzy najczęściej nie zadają sobie trudu, by wprowadzić czytelnika w zawiłości swoich specjalistycznych wywodów, nie biorą pod uwagę miejsca publikacji artykułu, tzn. charakteru naszego czasopisma.

Jeśli zaś chodzi o wymogi formalne, nie są duże. Co do objętości — maszynopis nie powinien przekraczać 12 znormalizowanych stron (30 wierszy z 60 znakami w wierszu). Jeśli jednak interesujący, zgodny z profilem pisma, temat wymaga większej objętości, to oczywiście względy formalne nie stoją na przeszkodzie. Rysunki techniczne, dołączane do tekstu, są w redakcji przygotowywane przez wykwalifikowanego kreślarza, nie muszą być zatem dostarczane na kalce; ważne natomiast, aby były całkowicie czytelne, przejrzyste i zachowujące konieczne proporcje.

Pożądany jest również dodatkowy materiał ilustracyjny (kontrastowe zdjęcia czarno-białe. wydruki, itp.), jeśli wzbogaca on bądź precyzuje wywód autora.

Maszynopis powinien być nadsyłany w dwóch egzemplarzach (oryginał plus kopia), z załączeniem aktualnych adresów, a także numerów telefonów, umożliwiających szybki kontakt. Autorów artykułów problemowych, którzy nie zamieszczali swych prac w ciągu dwóch ostatnich lat, prosimy ponadto o załączenie zdjęcia oraz życiorysu.

Autorzy, którzy chcieliby nawiązać bliższy kontakt z Czytelnikami, proszeni są o zamieszczenie adnotacji, iż godzą się na podanie na naszych łamach swojego adresu bądź telefonu.

REDAKCJA

NACZELNA ORGANIZACJA TECHNICZNA

WYDAWNICTWO SIGMA

CZASOPISM I KSIĄŻEK TECHNICZNYCH

ul. Świętokrzyska 14a
00-950 Warszawa
skrytka pocztowa 1004

KOLEGIUM REDAKCYJNE

Redaktor naczelny: prof. dr hab. Leon ŁUKASZEWICZ

mgr inż. Zbigniew GLUZA, dr Janusz GWIAZDA, Władysław KLEPACZ (zastępca redaktora naczelnego), dr inż. Tomasz PAWLAK, mgr Marek SOBCZYK, mgr Andrzej SZAŁAS, mgr Zbigniew ŚWIRSKI, dr inż. Janusz ZALEWSKI

Sekretarz redakcji: mgr Teresa JABŁOŃSKA

RADA PROGRAMOWA

Prof. dr hab. Tadeusz PECHE (przewodniczący), mgr inż. Tomasz BAŃKOWSKI (sekretarz), mgr inż. Antoni BOSSOWSKI, mgr inż. Roman BURNO, prof. dr hab. Andrzej JANICKI, mgr inż. Jan KRAMARCZUK, prof. dr hab. inż. Juliusz KULIKOWSKI, prof. dr hab. Leon ŁUKASZEWICZ, prof. dr hab. Antoni MAZURKIEWICZ, gen. dr inż. Marian PASTERNAK, dr inż. Bronisław PIWOWAR, mgr Zbigniew SUBSTYK, prof. dr hab. Tadeusz WALCZAK

Materiałów nie zamówionych Redakcja nie zwraca.

Redakcja: 00-041 Warszawa, ul. Jasna 14/16, pok. 243 i 244, tel. 27-71-40, dyżury redakcji 10.00—12.00

Zakł. Graf. „Tamka". Zam. 2130. Obj. 5,0 ark. druk. Nakład 3550 egz. M-94

Cena egzemplarza zł 75.— INDEKS 36124 Prenumerata roczna zł 900.—

9.1877/83

Informatyka

zastosowania w gospodarce, technice i nauce

Nr 6
MIESIĘCZNIK
1983
ROK XVIII
Czerwiec

ORGAN KOMITETU INFORMATYKI MINISTERSTWA NAUKI, SZKOLNICTWA WYŻSZEGO I TECHNIKI ORAZ KOMITETU NAUKOWO-TECHNICZNEGO NOT DS. INFORMATYKI

W NUMERZE: Strona

Programowanie współbieżne w systemie OS/JS na poziomie PASCALA
Andrzej Huzar 2

Projektowanie systemu graficznego. Część 1
Jacek Owczarczyk 5

Przegląd mikroprocesorów 16-bitowych (4). TEXAS INSTRUMENTS 9900
Jerzy Grabowski 8

Raport dla Klubu Rzymskiego (2)
— Wpływ komputeryzacji na miejsce pracy
John Evans 11

Standaryzacja języka BASIC (2). Moduł rdzeniowy języka
Wacław Iszkowski, Marek Maniecki 16

Prefiksowanie klasami w LOGLANIE
Hanna Oktaba 20

Sprzęgi systemowe w urządzeniach mikrokomputerowych
Mariusz Postół, Jerzy Staniaszczyk, Wojciech Weber, Andrzej Zdanowski 24

Pamięci dyskowe typu Winchester
Oprac. Maria Waśniewska 26

ALGORYTMY

Generowanie liczb pseudolosowych
Andrzej Szałas, Zbigniew Swirski 28

Z KRAJU

Informatyka w handlu
Maria Jerczyńska 31

GIEŁDA INFORMACJI 31

ZE ŚWIATA

Lawina komputerów domowych
Oprac. Marek Sobczyk 33

Rynek 1982. Mikroprocesory i pamięci
Oprac. Marianna Sobczyk 34

Superkomputery w Hollywood (*MkS*) 35

CSNET jednoczy informatyków (*MkS*) 36

TERMINOLOGIA

Słownictwo z zakresu inżynierii oprogramowania
Janusz Zalewski 37

W SKRÓCIE 38

POGLĄDY

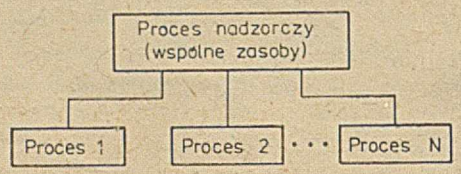
Dylemat „skłóconych"
Janusz Gwiazda III okł.

ANDRZEJ HUZAR
Centrum Obliczeniowe
Politechnika Wrocławska

Programowanie współbieżne w systemie OS/JS na poziomie PASCALA

W ostatnich latach wzrosło zainteresowanie programowaniem równoległym. Pojawiło się wiele publikacji mówiących zarówno o językach programowania współbieżnego, jak i o sposobach ich implementacji (p. [2], gdzie znajduje się bogata bibliografia). Istniejące języki programowania współbieżnego, takie jak ADA, CONCURENT PASCAL, CLAN, EDISON czy MODULA, zostały uznane jako odpowiednie narzędzia do rozwiązywania problemów programowania współbieżnego. Jednakże mają one dla ewentualnego użytkownika w Polsce istotną wadę — nie ma dostępnych implementacji tych języków. Okazuje się jednak, że w oparciu istniejące oprogramowanie komputera R-32 można, dopisując odpowiednie niewielkie procedury, przystosować język PASCAL do programowania równoległego. Niniejszy artykuł przedstawia główne idee takiego postępowania i niektóre szczegóły implementacyjne.

Podstawowym pojęciem w tej implementacji jest proces sekwencyjny (proces). Procesem nazwijmy każdy realizowany ciąg obliczeń według pewnego programu napisanego w PASCALU. Każdy istniejący proces ma do swej dyspozycji procesor wirtualny. Proces może zostać zawieszony, tzn. jego procesor wirtualny może wstrzymać swe działanie lub też może on spowodować odwieszenie innego zawieszonego procesu. Procesy mogą korzystać ze wspólnych zasobów. Jednakże w celu zagwarantowania poprawnej i kontrolowanej pracy procesów zakłada się, że w skończonym przedziale czasowym tylko jeden proces może korzystać z określonego zasobu. Wynika z tego, że musi istnieć mechanizm synchronizacji zapewniający spełnienie warunków korzystania z zasobów.



Rys. 1

Zrealizowana implementacja posiada strukturę przedstawioną na rysunku 1.

Proces nadzorczy służy definiowaniu wspólnych zasobów, inicjuje pozostałe procesy, a następnie realizuje zadania przydziału zasobów. W procesie nadzorczym zdefiniowane są również wspólne zasoby służące celom synchronizacji. Pozostałe procesy mając dostęp do wspólnych zasobów, mogą z nich korzystać tylko na żądanie. Służy do tego celu procedura standardowa PRZYDZIEL, za pomocą której można również zwalniać uprzednio zajęte zasoby. Tak więc zrealizowano tu mechanizm regionów krytycznych. Poza trzema krótkimi procedurami w asemblerze, całość została zrealizowana w PASCALU OS/JS (kompilator PASCALA F).

SPOSÓB UZYSKANIA WSPÓŁBIEŻNOŚCI I SYNCHRONIZACJI W PASCALU

Możliwość programowania współbieżnego uzyskano w oparciu o następujące cechy kompilatora PASCALA F oraz systemu operacyjnego OS/JS:

- kompilator umożliwia dołączanie programu napisanego w asemblerze
- kompilator umożliwia oddzielne kompilowanie procedur napisanych w tym języku oraz ich dołączanie do programów napisanych w asemblerze
- na poziomie asemblera możliwe jest:
  — zawieszenie procesu
  — wznowienie działania innego zatrzymanego procesu
  — powołanie nowego procesu.

Wykorzystując te możliwości zostały napisane trzy procedury w asemblerze, których funkcje są następujące:

- PROCEDURE WAIT (VAR A:ECB); EXCALL;

Wywołanie procedury WAIT(A) powoduje zawieszenie wykonywania procesu do momentu odwieszenia tego procesu przez inny proces. Jeśli sygnał wznawiający proces (patrz procedura POST) został wydany przed wykonaniem WAIT, to proces nie zostaje zawieszony. Aby proces mógł być ponownie zawieszony, musi być nadana odpowiednia wartość zmiennej A. Typ ECB jest pewnym typem definiowanym standardowo, a EXCALL słowem kluczowym oznaczającym procedurę kompilowaną poza ciałem programu.

- PROCEDURE POST (VAR A:ECB); EXCALL;

Wywołanie procedury POST(A) powoduje albo wznowienie pracy innego procesu, który wykonał procedurę WAIT na tej samej zmiennej A, albo — jeśli inny proces wykona później WAIT na zmiennej A — to nie ulegnie on zawieszeniu.

- PROCEDURE PROCES (VAR NA:NAZWA;VAR K:BLOK MONITORA;
VAR A:STRBLOK; ...); EXCALL;

Wywołanie procedury PROCES (NA,K,A,...) powoduje powołanie nowego procesu o nazwie równej wartości zmiennej NA (NAZWA jest typem ARRAY (1...8) OF CHAR, zmienne K, A służą potrzebom synchronizacji, a dalsze miejsca zajmują parametry reprezentujące wspólne zasoby). Powołany proces działa według procedury skompilowanej oddzielnie o następującym nagłówku:

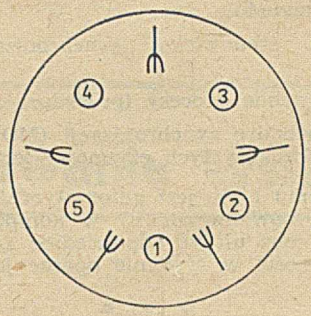
PROCEDURE „NA" (VAR K:BLOKMONITORA;VAR A:STRBLOK;...) gdzie „NA" jest wartością zmiennej NA.

(Dokładny opis działania makroinstrukcji programu nadzorczego systemu OS/JS POST, WAIT, ATTACH zawarty jest w [1], [3]). W oparciu o przedstawione procedury został zaimplementowany mechanizm regionów krytycznych. Implementacja ta została napisana już tylko w PASCALU.

PRZYKŁAD PROGRAMU WSPÓŁBIEŻNEGO

Dla ilustracji opisywanego mechanizmu przedstawimy rozwiązanie znanego w literaturze problemu filozofów [2]. Oto dokładne jego sformułowanie:
Przy okrągłym stole siedzi L FILOZOFÓW (na rys. 2 L = 5). Przed każdym z nich stoi talerz, na którym znajdu-

je się spaghetti. Pomiędzy każdymi dwoma talerzami leży widelec.



Rys. 2

Każdy z filozofów może:

— myśleć — wtedy filozof siedzi i nie zajmuje żadnego z widelców

— jeść — wtedy filozof trzyma w rękach, po jednym (w każdej dłoni) ze znajdujących się koło niego widelców.

Obowiązują jednak następujące zasady:

— jeśli filozof chce jeść, to może rozpocząć jedzenie wtedy, gdy dwa sąsiednie widelce są wolne; jeśli tak nie jest, to filozof czeka na zwolnienie tych widelców przez swoich kolegów

— jeśli filozof skończył jedzenie, to kładzie na stół dwa zajmowane przez siebie widelce; od tej pory filozof jest w stanie myślenia, a odłożone widelce mogą być zajęte przez innych filozofów

— zarówno w stanie myślenia, jak i w stanie jedzenia filozof znajduje się w skończonym czasie.

Problem polega na tym, aby żaden z filozofów nie uległ zagłodzeniu, tzn. aby po skończonym czasie uzyskał dostęp do widelców, jeśli zechce jeść.

Rozwiązanie tego problemu w zaimplementowanym systemie jest bardzo łatwe. Program, według którego działa każdy proces filozofa, jest następujący:

```
PROGRAM FILOZOF;
  CONST
       MAXPROCESOW=31;MAXZASOBOW=31;
TYPE BLOKMONITORA=RECORD
                    {standardowe pola}
                    END;
       STRBLOK=RECORD
                    {standardowe pola}
                    END;
       ZASOBY=SET OF 0..MAXZASOBOW
  VAR I,J, LICZBA:INTEGER;
PROCEDURE FILOZOF (VAR K:BLOKMONITORA,VAR
                                        A:STRBLOK);
PROCEDURE PRZYDZIEL (VAR K:BLOKMONITORA;VAR
                    A:STRBLOK); C:ZASOBY) ; EXCALL;
PROCEDURE CZEKAJ (VAR T:INTEGER) ; EXCALL;
PROCEDURE LOSUJ (VAR I:INTEGER);
  BEGIN I:= TIME MOD 300 END;
PROCEDURE JE;
  BEGIN LOSUJ (I); CZEKAJ (I) END;
PROCEDURE MYSLI;
  BEGIN LOSUJ(I); CZEKAJ (I) END;
BEGIN
1 PRZYDZIEL(K,A,(/0/);K.LIP:=K.LIP+1;PRZYDZIEL(K,A,(/ /);
    {SYNONIMAMI [,] SA W PASCALU F (/,/)
2 LOSUJ(LICZBA;LICZBA:=LICZBA MOD 10;
3 FOR I:=1 TO LICZBA DO
4 BEGIN MYSLI;PRZYDZIEL (K,A,(/A↑NR/)+
  (/ (A↑.NR+1) MOD K.LZASOB/)); JE;PRZYDZIEL (K,A,(/ /))
5 END;
6 PRZYDZIEL (K,A,(/0/) ;K.LIP:=K.LIP—1;PRZYDZIEL (K,A,(/ /)
  END
```

Znaczenie napisanych procedur jest następujące:

● Wywołanie PRZYDZIEL (K,A,C) powoduje przydział podzbioru zasobów reprezentowanych przez zmienną C; gdy C jest zbiorem pustym, to następuje zwolnienie wszystkich uprzednio zajętych zasobów. Każdy zasób jest reprezentowany przez liczbę całkowitą z przedziału 1..K.LZASOB — rola zasobu zerowego jest inna — służy sygnalizowaniu, że dany proces rozpoczął lub zakończył działanie. Instrukcje zawarte w liniach 1 i 6 są standardowe dla każdego procesu. W tym przykładzie zasób jest abstrakcyjny — nie jest reprezentowany przez żadną zmienną. Proces, aby korzystać z określonego zbioru zasobów, musi wpierw zażądać jego przydziału. Sekwencja PRZYDZIEL (K,A,C1); (dowolne instrukcje nie zawierające wywołania PRZYDZIEL/ PRZYDZIEL (K,A,C2), gdzie C1, C2 niepuste, jest równoważna sekwencji PRZYDZIEL (K,A,C1); ... ; PRZYDZIEL(K,A,(//);PRZYDZIEL(K,A,C2). Procedura PRZYDZIEL napisana jest w PASCALU i korzysta z procedur POST i WAIT.

● Wywołanie CZEKAJ(I) powoduje zawieszenie procesu na okres I setnych sekundy czasu rzeczywistego. Procedura napisana jest w asemblerze i służy symulacji upływu czasu.

● Wywołanie LOSUJ(I) powoduje nadanie zmiennej I losowej wartości z przedziału 0-299 — zmienna I jest potem parametrem procedury CZEKAJ.

● Procedury JE i MYSLI służą symulacji takich stanów u filozofów. Dalsza część programu jest zrozumiała, poza wywołaniem procedury PRZYDZIEL w wierszu 4. Wykorzystano tu fakt, że z każdym procesem związany jest rekord zawierający pole, którego wartością jest numer danego procesu. Natomiast w rekordzie K, który jest wspólny dla wszystkich procesów w polu LZASOB zawarta jest informacja określająca liczbę zasobów. W tym programie liczba zasobów jest równa liczbie procesów. Filozof cykl jedzenia i myślenia wykonuje losową liczbę razy — pętla FOR.

Treść programu nadzorczego jest następująca:

```
PROGRAM NADZORCA (INPUT,OUTPUT);
CONST MAXPROCESOW=31,MAXZASOBOW=31;
TYPE   BLOKMONITORA= {standardowe pola}
       STRBLOK      =
       TABLICA=ARRAY(/1.MAXPROCESOW/) OF STRBLOK;
       NAZWA=ARRAY (/1..8/) OF CHAR;
VAR K:BLOKMONITORA;T:ARRAY(/..MAXPROCESOW/) OF
                                              STRBLOK;
    NA:NAZWA;
    I,LZASOBOW,LPROCESOW:INTEGER;
PROCEDURE STARTS (VAR K:BLOKMONITORA;VAR T:TABLICA;
LZASOBOW,PROCESOW:INTEGER);EXCALL;
PROCEDURE MONITOR (VAR K:BLOKMONITORA;VAR
                                        T:TABLICA;
LZASOBOW,LPROCESOW:INTEGER) ;EXCALL;
PROCEDURE PROCES(VAR NA:NAZWA,VAR:K:BLOKMONITORA;
                                    VAR A:STRBLOCK)
EXCALL;
BEGIN
READ(LZASOBOW,LPROCESOW);
STARTS(K,A,LZASOBOW,LPROCESOW);
NA:='FILOZOF';
FOR I:=1 TO LPROCESOW DO PROCES (NA,K,A);
MONITOR(K,A,LZASOBOW,LPROCESOW)
END.
```

Znaczenie napisanych procedur jest następujące:

● Procedura STARTS powoduje utworzenie odpowiednich pól potrzebnych do synchronizacji i nadaje wartości początkowe tym polom — napisana jest wyłącznie w PASCALU.

● procedura MONITOR zarządza przydziałem żądanych zasobów — napisana jest w PASCALU, ale korzysta z procedur POST i WAIT

● procedura PROCES została opisana poprzednio.

ZASADY PRZYDZIAŁU ZASOBÓW

Jak wspomniano wcześniej, przydziałem zasobów steruje procedura MONITOR. Zrealizowano w niej następujący algorytm przydziału zasobów.

Krok A. Jeżeli jakiś proces zażąda przydziału zasobów, to zostaje on włączony na koniec kolejki procesów oczekują-

cych na zasoby, natomiast jeżeli jakiś inny proces żąda zwolnienia zasobów, to zasoby te są natychmiast zwalniane. Krok A jest wykonywany dla każdego procesu.

Krok B. W tym kroku rozpatrywana jest kolejka procesów oczekujących na przydział zasobów. Proces, który znajduje się na pierwszym miejscu w kolejce otrzyma żądane zasoby wówczas, gdy będą one wolne. Jeżeli część zasobów, na które oczekuje pierwszy w kolejce proces, jest wolna, to te zasoby nie zostaną zajęte przez inne procesy. W wyniku stosowania tej zasady, pierwszy w kolejce proces otrzyma w skończonym czasie żądane zasoby (bo mogą one być tylko przez skończony czas zajęte przez inne procesy). Procesowi następnemu w kolejce będą przyznane żądane zasoby, jeśli nie są one zajęte przez inne procesy i nie czeka na nie (lub na ich część) proces, który jest pierwszy w kolejce.

Kroki A i B są wykonywane kolejno w pętli, do momentu, gdy wszystkie procesy zgłoszą, że zakończyły działanie. Zawsze po wykonaniu kroków A i B proces nadzorczy zostaje zawieszony. Zostaje on odwieszony, gdy zmienią się warunki przydziału zasobów, tzn. gdy określony proces zażąda zwolnienia bądź przydziału zasobów. Zawieszone są również procesy, które oczekują na przydział bądź zwolnienie zasobów. Proces nadzorczy wznawia działanie procesu, któremu przyznał zasoby lub zwolnił poprzednio zajęte zasoby.

Zrealizowany algorytm dostępu do zasobów ma następujące cechy:

— każdy proces w skończonym czasie otrzyma żądany zasób — wynika to z tego, że pierwszy w kolejce otrzymuje żądany zasób oraz z tego, że nowe procesy są włączane na koniec kolejki; tak więc każdy proces albo uzyska żądane zasoby, albo znajdzie się w skończonym czasie jako pierwszy w kolejce i wtedy w skończonym czasie uzyska żądane zasoby

— o miejscu w kolejce decyduje moment zgłoszenia przydziału zasobów

— tylko jeden proces może korzystać jednocześnie z konkretnego zasobu.

Można więc uznać, że ten algorytm przydziału zasobów jest sprawiedliwy — nie wyróżnia on bowiem żadnego z procesów. Niemożliwe jest również zakleszczenie ani „zagłodzenie" żadnego z procesów. W tym sensie można zrealizowany system uznać za odpowiedni do rozwiązywania problemów programowania współbieżnego rozwiązywalnych za pomocą regionów krytycznych.

Aby przetestować działanie procesu synchronizacji, do procedury MONITOR została dołączona procedura wydruku wszystkich istotnych informacji o stanie tego procesu.

Tak więc, aby skorzystać z przedstawionego systemu, programista musi:

- napisać treść własnych procesów, do których dołącza pewne standardowe obiekty, i musi je skompilować z opcją „PROCEDURE" kompilatora PASCAL F
- w programie nadzorczym:

— zdefiniować wspólne zasoby (nadać im odpowiednie numery)

— wczytać odpowiednie dane

— przygotować środowisko synchronizacji (procedura STARTS)

— powołać potrzebne procesy (procedura PROCES)

— wywołać procedurę synchronizacji (MONITOR).

Kolejność wykonania tych czynności jest istotna dla poprawności systemu i nie jest automatycznie sprawdzana — wymagałoby to bowiem zmiany w kompilatorze. Niezależnie od tego, trzeba ułożyć odpowiednie zadania dla realizacji tych czynności w systemie operacyjnym OS/JS.

* * *

Zrealizowana implementacja pokazuje, że nie dysponując kompilatorami języków programowania równoległego, można — stosując odpowiednią metodologię — programować równolegle w systemie OS/JS w warunkach pewnego komfortu — tzn. na poziomie PASCALA i bez zaprzątania uwagi sprawami synchronizacji. Wprawdzie zrealizowana implementacja regionów krytycznych nie obejmuje sporej klasy problemów programowania współbieżnego, ale dysponując procedurami POST, WAIT, PROCES można równie łatwo zrealizować synchronizację procesów wzajemnie komunikujących się. Można zastanawiać się nad efektywnością przedstawionej implementacji. Okazuje się, że jeden cykl pracy procedury MONITOR zajmuje 0.1—0.2 sekundy czasu procesora. Tak więc koszt użycia implementacji zwiększa się wraz ze zwiększaniem się liczby żądań przydziału i zwolnień zasobów. Jednakże wzrost ten jest wolniejszy niż wzrost liczby żądań, ponieważ w jednym cyklu działania programu nadzorczego mogą być zwykle realizowane żądania kilku procesów. Wydaje się, że omówiona implementacja nadaje się głównie do symulacji procesów rzeczywistych. Umożliwia ona bowiem prostą realizację tego, co w sekwencyjnym języku programowania wykonywane jest w sposób bardziej skomplikowany. Wadą implementacji jest natomiast to, że poprawne działanie systemu jest możliwe tylko przy sztywnym trzymaniu się sformułowanych poprzednio zasad programowania. Usunięcie tej niedogodności nie jest możliwe bez zmian w kompilatorze PASCALA F.

LITERATURA

[1] Głowacki M.: System operacyjny DOS i OS. WNT, Warszawa, 1982

[2] Iszkowski W., Maniecki M.: Programowanie współbieżne. WNT, Warszawa, 1982

[3] OS/IS Makroinstrukcje supervisora. WZE MERA-ELWRO. Wrocław, 1976

[4] PASCAL OS/JS. Opis języka. IMM, Warszawa, 1977

[5] PASCAL OS/JS. Przewodnik programisty. IMM, Warszawa

[6] Struble G.W.: Wprowadzenie do programowania w języku asemblera. WNT, Warszawa, 1982.

KONFERENCJE

Zastosowanie mikroprocesorów

Oddział Wielkopolski Towarzystwa Naukowego Organizacji i Kierownictwa przy współpracy z Komitetem Naukowo-Technicznym Oddziału Wojewódzkiego NOT w Poznaniu organizują w dniach 18—19 października w Kołobrzegu konferencję naukową nt. „Mikroprocesory — stan i perspektywy zastosowań w Polsce". Tematyka konferencji:

- sprzęt i oprogramowanie mikroprocesorów
- metodyka projektowania i uruchamiania systemów mikroprocesorowych
- wpływ technik mikroprocesorowych na rozwój sprzętu informatycznego, urządzeń automatyki, telekomunikacji oraz techniki pomiarów
- zastosowanie mikroprocesorów w systemach przetwarzania danych, do wspomagania badań naukowych i obliczeń inżynierskich oraz w systemach sterowania procesami technologicznymi
- polityka kształcenia kadr.

Organizatorzy zapraszają do zgłaszania referatów (komunikatów), których czterostronicowe streszczenia należy nadsyłać do 31 sierpnia 1983 r. Streszczenia referatów (komunikatów) zaakceptowanych zostaną opublikowane w materiałach konferencyjnych.

W czasie trwania konferencji będzie zorganizowana giełda oprogramowania mikrokomputerów. Organizatorzy apelują o jak najliczniejsze nadsyłanie ofert.

Uczestnictwo w konferencji należy zgłaszać do dnia 31 sierpnia 1983 r. pod adresem: Towarzystwo Naukowe Organizacji i Kierownictwa, Oddział Wielkopolski, ul. Składowa 12, 61-897 Poznań.

Opłata uczestnictwa wynosi 3107 zł od osoby (w tym koszt materiałów konferencyjnych — 481 zł). W ramach opłaty organizatorzy zapewniają zakwaterowanie i wyżywienie przez dwie doby.

JACEK OWCZARCZYK
Warszawa

Projektowanie systemu graficznego. Część 1

Coraz szersze zastosowania grafiki komputerowej i cyfrowego przetwarzania obrazów powodują, że w wielu ośrodkach zajmujących się informatyką zachodzi konieczność rozszerzenia istniejących konfiguracji sprzętu obliczeniowego o system graficzny. W sytuacji, gdy z pewnych względów (najczęściej finansowych) zakup gotowego systemu nie jest możliwy, alternatywnym rozwiązaniem jest opracowanie systemu graficznego we własnym zakresie. Należy przy tym pamiętać, że udana konstrukcja systemu graficznego (jakkolwiek w zamyśle jednostkowa) pozwoli podjąć produkcję małoseryjną, z kolei zysk i doświadczenie umożliwi opracowanie nowszego, doskonalszego modelu...

Obie części niniejszego artykułu (druga traktująca o projektowaniu podstawowych układów, ukaże się w następnym numerze INFORMATYKI) dotyczą projektowania rastrowych systemów graficznych, zgodnego z aktualnymi tendencjami w konstrukcji komputerowego sprzętu graficznego [3].

Ograniczenie się do problemów związanych z projektowaniem sprzętu wynika przede wszystkim z dwóch powodów. Po pierwsze — projektowanie oprogramowania systemów graficznych (a zwłaszcza oprogramowania użytkowego) jest częściej przedmiotem publikacji, w tym również opracowań książkowych, np. [1, 2]. Po drugie — oprogramowanie to różni się istotnie w zależności od przeznaczenia systemu graficznego, podczas gdy zarówno systemy przetwarzania obrazów, jak i systemy grafiki komputerowej (jeżeli mają organizację rastrową) wykazują wiele istotnych podobieństw układowych.

ZAŁOŻENIA WSTĘPNE

Czynnikiem warunkującym projektowanie systemu graficznego — jak zresztą każdego innego urządzenia — powinno być zapotrzebowanie użytkownika. To oczywiste stwierdzenie warte jest przypomnienia, gdyż projektowanie systemu bez znajomości wymagań użytkowników utrudnia wybór prawidłowych rozwiązań. Szczególnie, gdy system przewidziany jest do produkcji seryjnej, prace projektowe powinny być poprzedzone badaniami rynku.

Ustalenie parametrów technicznych, schematu funkcjonalnego, kryteriów oceny, wybór środków technicznych do realizacji urządzenia oraz rozdział zadań do realizacji układowej i programowej — to pierwsze czynności w projektowaniu każdego urządzenia komputerowego. W przypadku systemu graficznego należy ponadto uwzględnić otoczenie systemu, tzn. całokształt zagadnień związanych ze współpracą systemu graficznego z innymi systemami komputerowymi. W szczególności — możliwość korzystania z mocy obliczeniowej, pamięci masowych, urządzeń peryferyjnych (np. drukarek elektrostatycznych do produkcji trwałych kopii wyświetlanych obrazów) innych systemów komputerowych.

Asortyment środków technicznych, którymi obecnie dysponuje konstruktor systemów graficznych, jest bardzo szeroki. I wszystko wskazuje na to, że będzie coraz szerszy. Nowe typy układów scalonych nie są bowiem konstruowane jedynie z myślą o systemach graficznych, lecz przede wszystkim — są wynikiem prowadzonych równolegle badań nad telewizją cyfrową, której odbiornik będzie w istocie terminalem graficznym o organizacji rastrowej.

WYBÓR SYSTEMU MIKROPROCESOROWEGO

Z porównania aktualnie produkowanych systemów graficznych [4] wynika, że w większości przypadków zawierają one w swej konfiguracji system mikroprocesorowy. Z tego też względu projektowanie systemu graficznego można traktować jako projektowanie specjalizowanego systemu mikroprocesorowego. Przy wyborze procesora można więc korzystać z kryteriów stosowanych w metodach projektowania mikroprocesorowych systemów użytkowych. Analiza parametrów dostępnych systemów mikroprocesorowych powinna wskazać system najlepszy.

W praktyce wybór systemu mikroprocesorowego jest ponadto wypadkową wielu zależnych od siebie czynników, takich jak: koszt projektowanego urządzenia, czas realizacji projektu (niektóre firmy dostarczają gotowe pakiety procesora, układów pamięci, układów wejścia-wyjścia — co znacznie skraca czas realizacji projektu), przeznaczenia systemu (konstrukcja jednostkowa, czy produkcja seryjna). Bardzo ważne jest również doświadczenie i wyposażenie techniczne zespołu projektującego system graficzny, a w szczególności możliwość korzystania z systemów wdrożeniowych (takich jak np. MDS firmy INTEL czy EXORCISER firmy MOTOROLA), programów skrośnych, szczegółowej dokumentacji technicznej.

W większości aktualnie produkowanych systemów graficznych stosuje się systemy mikroprocesorowe z 16-bitową szyną danych. W systemach przetwarzania obrazów jest to najczęściej jeden z mikrokomputerów serii LSI—11 (DEC). Atrakcyjność tych procesorów polega na tym, że zachowują one wiele cech funkcjonalnych (lista rozkazów, sposoby adresowania) popularnych, również w kraju, komputerów serii PDP 11. Istnieje więc możliwość opracowywania oprogramowania na komputerach, które dysponują dużym zestawem środków do uruchamiania i testowania programów.

W systemach grafiki komputerowej konstrukcja specjalizowanego procesora opartego na mikroprocesorach modułowych zwiększa szybkość systemu, ale wymaga odpowiedniego doświadczenia (i zazwyczaj czasu). Jeżeli szybkość systemu nie jest czynnikiem decydującym, to alternatywnym rozwiązaniem jest użycie typowego systemu mikroprocesorowego. W małych systemach (terminalach graficznych) mogą być również stosowane systemy mikroprocesorowe z 8-bitową szyną danych, zwłaszcza jeżeli w konstrukcji urządzenia wykorzystany jest jednostkowy graficzny kontroler wizyjny (opisany w drugiej części artykułu).

WYMIARY RASTRA

Jakkolwiek wymagania użytkownika i planowane zastosowania systemu graficznego pozwalają oszacować wymiary rastra, to przy ich ustalaniu należy ponadto uwzględnić następujące czynniki:

- Stosowanie standardu telewizyjnego (a stosowanie standardów zawsze jest wskazane) narzuca pewne ograniczenia w maksymalnych wymiarach rastra; np. dla standardu 625 linii jedynie 575 linii jest wyświetlanych na ekranie (pozostałe są wygaszane w czasie pionowego powrotu plamki). Biorąc pod uwagę, że stosunek szerokości do wysokości ekranu standardowego monitora telewizyjnego jest 4:3, każda linia powinna składać się z 767 elementów obrazu (punktów). A zatem dla tego standardu maksymalny wymiar rastra jest 767×575 punktów.

● Monitory telewizyjne posiadają na brzegach ekranu gorszą rozdzielność, dlatego też wskazane jest pozostawienie marginesów z każdej strony (w telewizji użytkowej przy zobrazowaniu informacji graficznej wynoszą one ok. 10% maksymalnych wymiarów rastra [8].

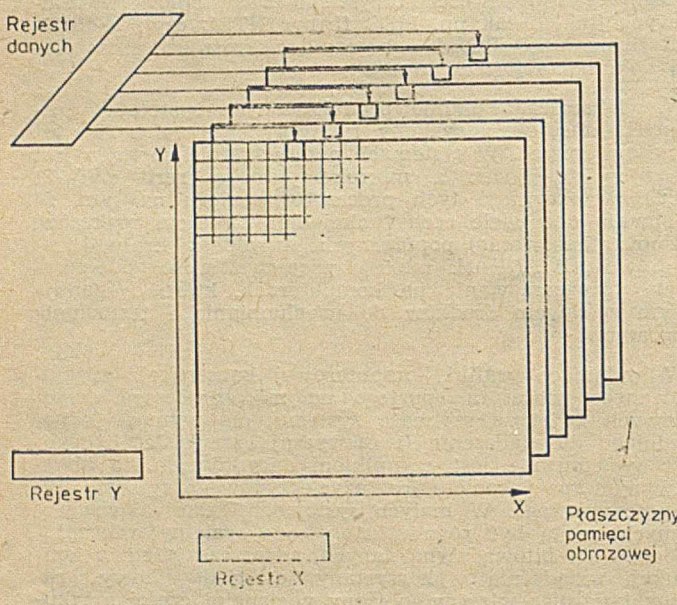
● Stosowanie większych (niż 767×575) wymiarów rastra powoduje konieczność użycia specjalnych (a więc droższych) monitorów, również konstrukcja kontrolera wizyjnego jest bardziej skomplikowana i wymaga bardzo szybkich układów.

● Wymiary rastra dla znaków alfanumerycznych, jeżeli są one zapisywane w odrębnej płaszczyźnie pamięci obrazowej, mogą być większe niż wymiary rastra dla informacji graficznej; umożliwi to wyświetlanie napisów również poza obrazem.

W średnich systemach graficznych (ang. medium scale resolution) stosowany jest zazwyczaj raster 512×512 punktów. W niektórych zastosowaniach (np. w medycynie) rastry są mniejsze — zwykle 256×256 punktów. Wymiary te, w przypadku gdy systemem graficznym zarządza mikroprocesor z 8-bitową szyną danych, mają dodatkową zaletę — do adresowania punktów w obu kierunkach wystarcza bowiem 8 bitów i rejestry adresujące mogą być ustawiane w pojedynczym cyklu.

PAMIĘĆ OBRAZOWA

Pamięć obrazowa w systemach graficznych o organizacji rastrowej może być traktowana jako zbiór jednobitowych płaszczyzn (rys. 1). Jakkolwiek wymiary pojedynczej płaszczyzny pokrywają się często z wymiarami rastra, to taka zgodność nie jest konieczna. Szczególnie — większe wymiary płaszczyzn umożliwiają tworzenie większych obrazów, które będą wyświetlane na ekranie monitora w częściach pokrywających się z wymiarami rastra.



Rys. 1. Organizacja i adresowanie pamięci obrazowej (każdemu punktowi rastra odpowiada 6-bitowa komórka pamięci obrazowej)

Istnieje oczywisty związek pomiędzy liczbą płaszczyzn i liczbą poziomów szarości (czy kolorów), które mogą być wyświetlane w poszczególnych punktach rastra. A zatem przeznaczenie systemu określa liczbę płaszczyzn pamięci obrazowej. W systemach grafiki komputerowej wystarcza na ogół od jednej (wyświetlanie jasnych elementów rysunku na ciemnym tle — lub odwrotnie) do trzech (8 kolorów lub poziomów szarości). W systemach przetwarzania obrazów wymagane są co najmniej cztery płaszczyzny (16 poziomów szarości).

Do konstrukcji bloku pamięci obrazowej można wykorzystać pamięci półprzewodnikowe RAM dynamiczne lub statyczne. Monolity pamięci dynamicznych ze względu na większy stopień upakowania są tańsze, ale wymagają specjalnych impulsów odświeżających (elementem utrzymującym informację jest tu pojemność bramek tranzystorów), co — pomijając konieczność stosowania dodatkowych układów odświeżających — wprowadza pewne ograniczenia w okresach dostępu do pamięci obrazowej.

Pamięci statyczne, stosowane często w konstrukcjach jednostkowych systemów graficznych (np. [5, 6]) są droższe, jakkolwiek wydaje się, że malejąca różnica w cenie (w porównaniu z pamięciami dynamicznymi) może doprowadzić do stosowania pamięci statycznych również w systemach produkowanych seryjnie. Wprowadzone przez firmę TOSHIBA (a także HITACHI) monolity pamięci statycznych o pojemności 16 K bitów wykonane technologią CMOS (o małym poborze mocy) załamały w 1981 roku ceny pamięci statycznych — kosztowały wówczas w cenie hurtowej poniżej 10$ za sztukę. Obecnie sprzedawane są w cenie detalicznej poniżej 6$.

Adresowanie pamięci obrazowej

Istnieje naturalna odpowiedniość pomiędzy poszczególnymi punktami rastra i komórkami pamięci obrazowej. Z tego względu najwygodniej jest użytkownikowi (programiście) systemu graficznego adresować poszczególne komórki pamięci obrazowej za pomocą dwóch rejestrów: X i Y (rys. 1). Trzeci rejestr służy do ustawiania wartości w poszczególnych komórkach, które później są zamieniane przez układy kontrolera wizyjnego na ustalone poziomy szarości czy barwy. W systemie mikroprocesorowym zarządzającym systemem graficznym rejestry te wchodzą w przestrzeń adresową dla układów wejścia-wyjścia. Ten sposób adresowania pamięci obrazowej posiada również pewną wadę — konieczność ustawiania trzech rejestrów wpływa hamująco na szybkość przesyłania danych do pamięci obrazowej.

Z punktu widzenia organizacji systemu mikroprocesorowego pamięć obrazowa może być również włączona w przestrzeń adresową układów pamięci mikroprocesora. Zaletą tego rozwiązania, popularnego w małych terminalach graficznych, jest możliwość traktowania pamięci obrazowej jak innych bloków pamięci i używania standardowego zbioru instrukcji przy zapisie (odczycie) danych. W konsekwencji jednak zmniejszony zostaje obszar pamięci, który może być przeznaczony na inne cele (programy, dane). W zależności od obszaru adresowego (liczby linii adresowych) ograniczona jest również wielkość pamięci obrazowej. Na przykład — dla typowego systemu mikroprocesorowego z 8-bitową szyną danych szyna adresowa składa się z 16 linii, a więc istnieje możliwość adresowania 64 K bajtów. Jedna płaszczyzna pamięci obrazowej dla rastra 512×512 punktów wymaga 32 K bajtów, co oznacza, że jest to praktycznie maksymalna wielkość pamięci obrazowej dla tego systemu.

W rozbudowanym systemie mikroprocesorowym z układem zarządzania pamięcią (takim jak np. LSI—11/23) istnieje możliwość adresowania pamięci obrazowej stronami, co pozwala na szybkie i elastyczne wypełnianie całych obszarów pamięci obrazowej.

Dostęp do pamięci obrazowej

W systemie graficznym dwa urządzenia muszą mieć dostęp do pamięci obrazowej: procesor — w celu aktualizacji jej zawartości (zmian wyświetlanego obrazu) oraz kontroler wizyjny, który odczytując kolejne bajty (słowa) generuje z nich sygnał wizyjny, przy czym częstotliwość jego dostępu wynika z zależności czasowych sygnału wizyjnego. Ponieważ fizycznie dostęp obu tych urządzeń nie może odbywać się jednocześnie, należy określić sposób bezkonfliktowej komunikacji z pamięcią obrazową. Nie licząc mało eleganckiej metody, w której procesor ma zawsze pierwszeństwo i w czasie jego dostępu kontroler wizyjny jest odłączony (co powoduje chwilowy zanik sygnału wizyjnego i migotanie obrazu), możliwe są tu dwa rozwiązania problemu:

1. Procesor uzyskuje dostęp do pamięci obrazowej tylko w czasie powrotu plamki (poziomego i pionowego), a więc wtedy, gdy kontroler nie generuje informacji wizyjnej i może być odłączony od pamięci obrazowej. Stan taki jest sygnalizowany za pomocą przerwań do procesora. Ponieważ jednak powroty plamki stanowią jedynie ok. 25% czasu wymaganego do wyświetlenia jednego kadru, rozwiązanie to ogranicza szybkość aktualizacji pamięci obrazowej (szybkość zmian wyświetlanego obrazu).

2. Procesor uzyskuje dostęp do pamięci obrazowej również w trakcie wybierania (roboczego ruchu plamki). Rozwiązanie to wymaga jednak użycia szybszych pamięci i wykorzystuje następującą zależność generowania sygnału wizyjnego: czas wyświetlania jednego punktu rastra wynosi ok. 80 ns; zakładając, że kontroler wizyjny pobiera jednocześnie grupę bitów dla kolejnych punktów rastra, np. 8 bitów, wówczas okres między kolejnymi zgłoszeniami kontrolera wynosi ok. 640 ns. Jeżeli pamięć obrazowa ma dostatecznie mały czas dostępu, wówczas okres ten można podzielić na dwie części, przeznaczając jedną z nich na dostęp kontrolera, a drugą — procesora. Kontroler wizyjny musi ponadto generować sygnał zezwalający na dostęp procesora jedynie w momencie, gdy pozostało wystarczająco dużo czasu na prawidłowy odczyt (zapis). Powyższe rozwiązanie spotykane jest również w terminalach alfanumerycznych [7].

Wyświetlanie znaków alfanumerycznych

Informacja graficzna wyświetlana na ekranie monitora musi być uzupełniana różnego typu pomocniczymi opisami. Wyświetlanie znaków alfanumerycznych można zrealizować programowo lub układowo, przy czym w obu wypadkach informacja o graficznym rozłożeniu punktów tworzących poszczególne znaki przechowywana jest w postaci matryc (rys. 2). W realizacji programowej matryce znaków zapisane są w pamięci mikroprocesora, a odpowiedni program przepisuje ich zawartość w żądane miejsca pamięci obrazowej. Umożliwia to dowolne położenie znaków na rastrze, a także — przez odpowiednie modyfikacje programu — zmianę orientacji czy wyglądu wyświetlanych znaków.

Znacznie szybsza realizacja układowa wykorzystuje zasadę generowania znaków, stosowaną w terminalach alfanumerycznych. Raster jest tu umownie dzielony na klatki (rys. 2), a jedna z płaszczyzn pamięci obrazowej służy do przechowywania informacji o znakach w poszczególnych klatkach. Matryce znaków przechowywane są w pamięci ROM (nazywanej tu również generatorem znaków) i są adresowane kodem (najczęściej jest to kod ASCII) znaków przechowywanych w pamięci obrazowej. Kontroler wizyjny generuje odpowiedni ciąg adresów (numer wiersza, numer kolumny i numer linii w wierszu) wypełniając znakami cały raster (rys. 2).

Płaszczyzna, w której magazynowana jest informacja o znakach alfanumerycznych, jakkolwiek funkcjonalnie jest częścią pamięci obrazowej, nie musi być adresowana w ten sam sposób jak pozostałe płaszczyzny pamięci. Z uwagi na stosunkowo niedużą wielkość (np. dla rastra 512×512 i matrycy 8×8 wymagane jest jedynie 4 K bajty) może być ona włączona w obszar układów pamięci mikroprocesora, co umożliwia używanie standardowego zbioru instrukcji do manipulowania znakami na ekranie monitora.

W zależności od repertuaru znaków, które mogą być wyświetlane na ekranie, na rejestrację ich kodów wystarcza 6 (64 znaki) lub 7 (128 znaków) bitów. Jeżeli w pamięci obrazowej każdej klatce odpowiada jeden bajt (8 bitów) wówczas pozostałe bity (bit) można wykorzystać jako bity atrybutu znaków. W zależności od konstrukcji kontrolera wizyjnego mogą one sterować odpowiednimi układami w celu uzyskania na ekranie określonych efektów, takich jak wyświetlanie znaku jako „uzupełnienie" punktów czy migotanie znaku.

LITERATURA

[1] Foley J. D., Van Dam A.: Fundamentals of interactive computer graphics. Addison-Wesley Pub., 1982

[2] Newman W. M., Sproull R. F.: Principles of interactive computer graphics. Mc Graw-Hill, 1979

[3] Owczarczyk J.: Rastrowe systemy graficzne. Informatyka nr 3, 1983

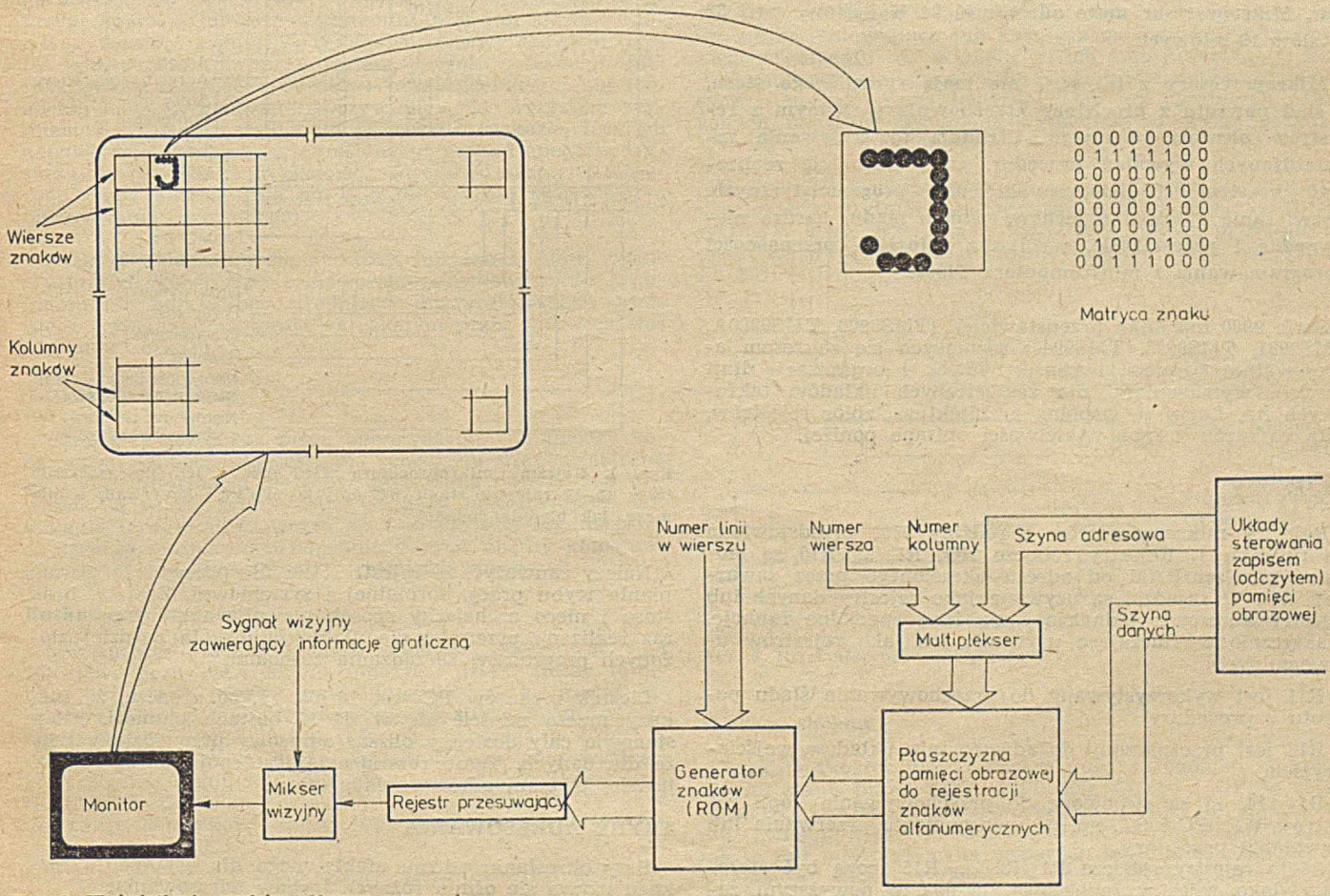
[4] Owczarczyk J.: Parametry użytkowe wybranych rastrowych systemów graficznych. Informatyka nr 4, 1983

[5] Prévot C.: Processeur du vision: realisation d'une memoire d'image. Minis et Micros, nr 159, 1982

[6] Takagi Y., Sumitani M., Yoshihara K.: High-speed image processor for time-resolved spectroscopy. Review of Scientific Instrumentation, vol. 52, nr 1, 1981

[7] Trottier L., Matic B.: Transparent memory ends conflicts over CRT control. Electronics, vol. 52, nr 14, 1979

[8] Walczyk J.: Podstawy systemów telewizji użytkowej, WKiŁ, 1982.



Rys. 2. Wyświetlanie znaków alfanumerycznych

JERZY GRABOWSKI
Warszawa

Przegląd mikroprocesorów 16-bitowych (4)
TEXAS INSTRUMENTS 9900

Mikroprocesory z serii 9900 były jednymi z pierwszych mikroprocesorów 16-bitowych. Ich architektura odpowiada architekturze procesora wcześniejszego minikomputera TMS990. Z mikroprocesorów przedstawionych w niniejszym cyklu są one najprostsze w sensie budowy i możliwości operacyjnych.

Mikroprocesor TMS9900 ma trzy rejestry zrealizowane w mikroukładzie, a funkcje właściwych rejestrów roboczych systemu mikrokomputerowego pełni szesnaście 16-bitowych słów umieszczonych w zapisywalnej pamięci RAM. Są to kolejne słowa pamięci określone przez bieżącą zawartość rejestru wskaźnika obszaru roboczego WP (ang. workspace pointer), oznaczane jako RO, R1..., R15 (należy pamiętać, że są to pary bajtów o adresach (WP)+0, (WP)+2..., (WP)+30). Układ TMS9900 posiada także licznik rozkazów PC oraz rejestr stanu ST o długości 16 bitów. Mikroprocesor może adresować 64 K bajtów, czyli 32 K słów 16-bitowych.

Mikroprocesory z tej serii nie mają sprzętowego stosu, a ślad powrotu z procedury przechowują w jednym z rejestrów obszaru roboczego. Utrudnia to stosowanie zagnieżdżonych wywołań procedur, gdyż trzeba je realizować kosztem specjalnych zabiegów programistycznych. Rozwiązanie z takim przechowywaniem śladu, bardzo niewygodnę i przestarzałe, wynika z założenia przenośności oprogramowania z minikomputera TI990.

Seria 9900 ma kilka przedstawicieli (TMS9900, TMS9980A, TMS9981, TMS9985, TMS9940) różniących się zakresem adresowalnej pamięci, liczbą końcówek i organizacją linii danych, wymaganiami dla zewnętrznych układów taktujących itp. Łączy je wspólna architektura, zbiór rejestrów, lista rozkazów i inne właściwości opisane poniżej.

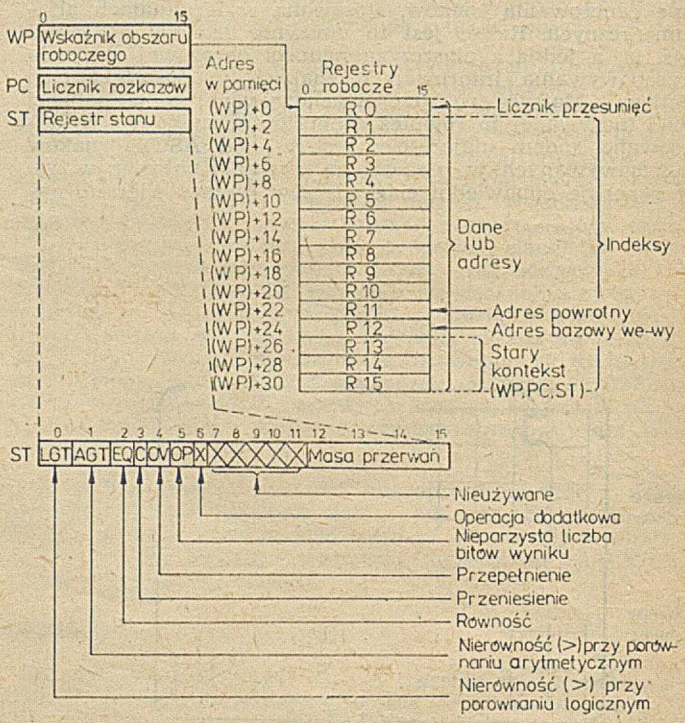
REJESTRY

Rejestry mikroprocesorów z rodziny 9900 przedstawiono na rysunku 1. Rejestry robocze RO, R1, ..., R15 są słowami pamięci RAM od adresu określonego przez zawartość WP. W zasadzie są używane jako rejestry danych lub rejestry adresowe, jednakże mają inne specjalne funkcje, praktycznie wykluczające używanie ich jako rejestrów uniwersalnych:

- R11 jest wykorzystywany do przechowywania śladu powrotu z procedury
- R12 jest przeznaczony do adresowania układów wejścia-wyjścia
- R13, 14, 15 są stosowane do przechowywania kopii rejestrów WP, PC i ST, np. po uaktywnieniu przerwania lub wywołania procedury.

Ponadto rejestry robocze R1, R2, ..., R15 mogą być stosowane jako rejestry indeksowe, jednak z powyższym zastrzeżeniem.

W rejestrze stanu ST wykorzystuje się 11 spośród 16 bitów. Sześć bitów najmniej znaczących (LGT, AGT, EQ, C, OV, OP) stanowi informację o wyniku rozkazu porównania liczb ze znakiem lub bez znaku oraz — o stanie przepełnienia i parzystości ostatniego wyniku operacji arytmetycznej. Bit X jest znacznikiem wykonywania tzw. kodu dodatkowego, czyli kodu nie zrealizowanego układowo, lecz obsługiwanego programowo. Bity 12—15 tworzą maskę przerwań.



Rys. 1. Rejestry mikroprocesora TMS 9900 i ich przeznaczenie (Bity 12—15 rejestru stanu ST opisują maskę przerwań, a nie masę, jak błędnie podano)

Należy zauważyć, że rejestr stanu nie przewiduje rozróżniania trybu pracy normalnej i systemowej. Zresztą, brak stosu i nieco archaiczny sposób wywoływania procedur oraz realizacji przerwań nie ułatwiają pisania bardziej złożonych programów zarządzania zasobami.

Licznik rozkazów PC jest 16-bitowy, co oznacza, że rozkazy muszą mieścić się w 64 K bajtach pamięci, które stanowią cały dostępny obszar adresowy przewidziany także dla danych. Takie rozwiązanie jest spójne z opisanymi poniżej trybami adresowania pamięci.

TRYBY ADRESOWANIA

Przy określaniu adresu efektywnego dla argumentu rozkazu używa się ośmiu różnych trybów adresowania:
1. rejestrowe — argumentem jest zawartość rejestru

2. rejestrowe pośrednie — adres efektywny jest określany przez zawartość rejestru

3. rejestrowe pośrednie z autoinkrementacją — adres efektywny jest określany przez zawartość rejestru, która następnie zostaje zwiększona o 2 lub 1 (odpowiednio dla operacji na słowach i bajtach)

4. bezpośrednie — adres efektywny jest zawarty w kodzie rozkazu

5. indeksowe — adres efektywny jest wyznaczany jako suma zawartości rejestru określonego rozkazem i bezpośredniego adresu również zawartego w kodzie rozkazu

6. względne — adres efektywny jest sumą zawartości licznika rozkazów PC i 8-bitowych przemieszczenia zawartego w kodzie rozkazu

7. proste — argument jest zawarty w rozkazie

8. specjalne — stosowane do adresowania urządzeń wejścia-wyjścia przez rejestr R12 (dokładniejszy opis podano przy omawianiu rozkazów wejścia-wyjścia).

Rozkazy mikroprocesora mają taki format, że użycie trybów 1—5 wyklucza pozostałe (i na odwrót). Omówiony zbiór trybów adresowania wydaje się rozsądnym kompromisem między funkcjonalnością a prostotą sprzętu.

LISTA ROZKAZÓW

Mikroprocesor TMS9900 ma 72 rozkazy podstawowe. Jest to liczba niewielka w porównaniu z innymi mikroprocesorami 16-bitowymi, a nawet w porównaniu z niektórymi 8-bitowymi. Warto jednak pamiętać, że minikomputer PDP—8, którego pojawienie się stworzyło nową epokę zastosowań techniki komputerowej miał tylko 8 podstawowych rozkazów. Do dziś jednak komputery z tej serii lub ich kopie wykorzystują ogromne oprogramowanie użytkowe dla nich napisane, a oprogramowanie podstawowe zawiera dyskowe systemy operacyjne, kompilatory języków i systemy obsługujące procesy w trybie nadążnym.

Rozkazy mikroprocesora TMS9900 składają się z jednego, dwóch lub trzech słów 16-bitowych. Pierwsze słowo określa operację, długość argumentu (bajt lub słowo 16-bitowe), rejestry argumentu źródłowego i wynikowego, tryby adresowania dla argumentów, licznik przesunięć (dla rozkazów przesunięć i obrotów) oraz przemieszczenie (dla adresowania względnego). Oczywiście, nie wszystkie wymienione elementy występują w każdym rozkazie, lecz ich odpowiednie kombinacje, określone przez kod operacji, występujący zawsze. Przykładowo — dla rozkazów stosujących tryby adresowania od 1—5 nie można użyć trybu 6 (adresowania względnego).

Rozkazy składające się z dwóch lub trzech słów wykorzystują drugie i trzecie słowo do przechowywania argumentu lub adresu bezpośredniego, interpretowanego zgodnie z trybem adresowania określonym przez pierwsze słowo kodu rozkazu.

Rozkazy przesyłania

Rozkazy z tej grupy mogą operować danymi między komórkami pamięci a rejestrami roboczymi oraz rejestrami ST i WP. Informacja może być przesyłana bajtami lub słowami. Są w tej grupie także rozkazy umożliwiające stosowanie trybów adresowania od 1—5 dla obu argumentów, co oznacza możliwość przesyłania danych między komórkami pamięci bez pośrednictwa rejestrów roboczych.

Rozkazy arytmetyczne

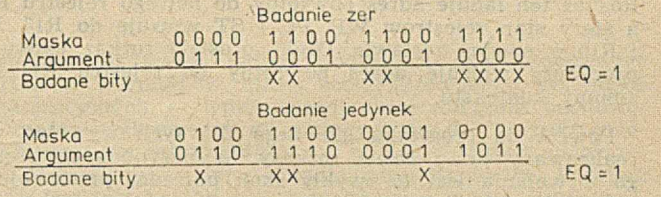
Mikroprocesor TMS9900 może dodawać, odejmować 8- i 16-bitowe liczby ze znakiem, mnożyć i dzielić liczby 16-bitowe bez znaku, uzyskując 32-bitowy iloczyn i stosując 32-bitową dzielną. Może też wykonywać automatyczną zmianę zawartości komórki lub rejestru roboczego o 1 lub 2, negację arytmetyczną oraz operację wyznaczania wartości bezwzględnej argumentu. Nie ma natomiast, rozkazów operacji na liczbach przedstawionych w kodzie BCD[1]) choć można je stosunkowo łatwo wykonywać przy użyciu innych rozkazów. Łącznie istnieje 13 rozkazów arytmetycznych, które stosują tryby adresowania 1—5 (i to dla obu argumentów, jeśli są przewidziane w rozkazie).

[1]) BCD — ang. binary coded decimal (zapis dziesiętny kodowany dwójkowo).

Rozkazy logiczne

Rozkazy z tej grupy dzielą się na rozkazy porównania, działania logiczne i przesunięcia.

Rozkazy porównań obejmują klasyczne porównania dwóch argumentów, ustawiające bity stanu w rejestrze ST. Argumentami mogą być zawartości rejestrów roboczych lub komórek pamięci określane przez tryby adresowania 1—5, a także argumenty bezpośrednie (tryb 7). Do tej grupy należą także rozkazy badania obecności zer lub jedynek na wybranych maską polach argumentu. Obecność jakiegokolwiek zera (jedynki) dla innego argumentu jest sygnalizowana przez ustawienie bitu EQ w słowie stanu ST. Na rysunku 2 przedstawiono przykłady działania tej operacji.

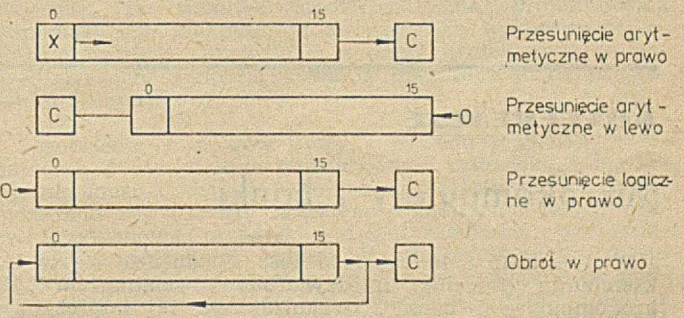


Rys. 2. Działania rozkazów badania argumentu na obecność zer i jedynek

Rozkazy działań logicznych obejmują klasyczny zestaw wyznaczania sumy, iloczynu, różnicy symetrycznej i zaprzeczenia logicznego. Osobliwością tego mikroprocesora jest ograniczenie narzucone na jeden z argumentów sumy i iloczynu logicznego, który może być tylko argumentem bezpośrednim (tryb 7).

Następną grupę tworzą rozkazy kasowania i ustawiania bitów. Podobnie jak w rozkazach badania obecności jedynek lub zer, jeden z argumentów jest interpretowany jako maska, ustawiająca lub kasująca odpowiednie bity drugiego argumentu bez względu na ich stan początkowy. Należy zauważyć, że rozkaz ustawiania bitów odpowiada klasycznej operacji sumy logicznej, lecz bez ograniczeń nałożonych na drugi argument wymienionego wyżej rozkazu wykonywania sumy logicznej.

Rozkazy przesunięć występują w czterech wariantach: przesunięcie arytmetyczne w prawo, przesunięcie arytmetyczne w lewo, przesunięcie logiczne w prawo, przesunięcie cykliczne w prawo. Ich działanie przedstawiono na rysunku 3.



Rys. 3. Działanie rozkazów przesunięć

Rozkazy skoków

Rozkazy skoków warunkowych służą do badania 12 kombinacji stanu bitów LGT, AGT, EQ, C, OV i OP w rejestrze ST. Dla skoków warunkowych stosuje się adresowanie względne z przemieszczeniem 8-bitowym. Przemieszczenie to jest interpretowane w parach bajtów, co oznacza, że warunkowo można wykonać skok w zakresie adresów od —256 do +255. Mikroprocesor nie ma rozkazów skoku z

badaniem i modyfikacją rejestru, tak wygodnych do organizowania wyliczanych iteracji. Można natomiast wykonać wyizolowany rozkaz wskazany przez adres efektywny rozkazu inicjującego wykonanie. Rozkaz ten (wyklinany przez niektórych autorów rozważających sensowność listy rozkazów różnych komputerów) stosowany bez ograniczeń umożliwia łatwe i szybkie pisanie programów niemożliwych do poprawienia. Natomiast używany z umiarem przydaje się przy pisaniu procedur uruchomieniowych.

W rozkazach skoków zwykłych stosuje się tryby adresowania 1—5, a więc można za ich pośrednictwem przemieszczać się w dowolne miejsce programu.

Mikroprocesor ma dwa rozkazy skoku do podprogramu. Każdy z nich przechowuje adres powrotny w rejestrze roboczym R11. Przy wywołaniu stosuje się różne tryby adresowania wywoływanej procedury. Jeden z rozkazów tzw. BLWP (ang. branch and load workspace pointer), umożliwia ustalenie nowego zbioru rejestrów roboczych, a więc wpisanie nowej zawartości do WP. Taką zmianę zbioru dostępnych rejestrów nazywa się przełączeniem kontekstu. Rozkaz ten ładuje adres powrotny do nowego rejestru R11, a stary stan rejestrów WP, PC i ST wpisuje do R13, R14 i R15 w nowym zestawie. Drugi, prostszy rozkaz, skoku do procedury wpisuje adres powrotny do rejestru R11 bez zmiany kontekstu.

Powrót do programu głównego dla tych wywołań jest realizowany przez różne rozkazy. W przypadkach prostego wywołania jest to zwykły skok pośredni przez rejestr, a w przypadku bardziej złożonym — specjalny rozkaz powrotu z odtworzeniem kontekstu. Jest to tzw. rozkaz RTWP (ang. return with workspace pointer), który odtwarza stan rejestrów WP, PC i ST według zawartości przechowanej w R13—R15, co powoduje jednocześnie powrót do programu głównego. Opis tych rozkazów jest dość zawiły, a wygoda przy ich stosowaniu raczej wątpliwa. Zagnieżdżone wywołanie procedur dla tego mikroprocesora doczekały się oddzielnej literatury opisującej stosowane do tego celu triki, a zapotrzebowanie na nią źle świadczy o przyjętym rozwiązaniu.

Rozkazy wejścia-wyjścia

Mikroprocesor TMS9900 ma specjalną technikę komunikacji z urządzeniami zewnętrznymi zwaną „metodą CRU" (ang. control register unit i/o, autor nie zna polskiego odpowiednika tego terminu. Technika ta polega na wymianie informacji z urządzeniem sposobem szeregowym, bit po bicie, tzn. adresuje się i przesyła pojedyncze bity, a nie słowa lub bajty. Metoda jest podobno elastyczna, a mikroprocesor ma specjalne rozkazy do komunikacji z urządzeniami sprzęganymi z systemem mikrokomputerowym.

Rozkazy wejścia-wyjścia wykorzystują rejestr roboczy R12 do przechowywania tzw. bazy do komunikacji z urządzeniami sprzęganymi tą techniką. Wymieniony rejestr określa adres najmniej znaczącego bitu z grupy 16 bitów (w większości przypadków bity są adresowane porcjami po 16). Mikroprocesor ma pięć rozkazów realizujących transmisję danych metodą CRU. Dwa z nich umożliwiają przesyłanie informacji do i z jednostki sprzęgającej. Argumenty zawierają liczbę bitów do przesłania, adres przeznaczenia lub źródła w rejestrach mikroprocesora i adres bazy w obrębie CRU przez zawartość R12. Pozostałe rozkazy wejścia-wyjścia umożliwiają ustawianie, kasowanie i testowanie adresowanych w ten sposób bitów.

Tak jak i dla innych procesorów, urządzenia zewnętrzne mogą być sprzęgane przez odwzorowanie części obszaru adresowego na rejestry urządzeń zewnętrznych (ang. memory mapped i/o). Wówczas dowolny rozkaz adresujący obszar pamięci przeznaczony dla urządzeń wejścia-wyjścia staje się rozkazem przesyłania zewnętrznego. Oczywiście, specjalizowane rozkazy CRU nie są wtedy używane ani obsługiwane przez układy sprzęgające.

PRZERWANIA

Mikroprocesor TMS9900 jest przystosowany do obsługi przerwań priorytetowych na 16 poziomach. Przerwania są maskowalne. Do ustalania stanu maski przerwań, przechowywanej w rejestrze ST, służy specjalny rozkaz stosujący argument bezpośredni (adresowanie proste). Jest to jedyny rozkaz kontroli układów przerwań.

Przerwanie jest wymuszane przez akcję analogiczną do opisanego powyżej rozkazu BLWP, wskutek czego tworzy się nowy zbiór rejestrów roboczych, a stary stan WP, PC i ST zostaje przechowany w rejestrach R13, R14 i R15 nowego zestawu. Sekwencja przerwania jest następująca:

1. Urządzenie zgłasza przerwanie przez stosowną linię.

2. Czterema liniami przesyłany jest priorytet urządzenia określający jednocześnie wektor przerwania.

3. Nowe wartości PC i WP są pobierane spod adresu określonego wektorem, co oznacza zainicjowanie rozkazu BWLP i przełączenie kontekstu.

4. Stare wartości PC, WP i ST są przechowywane w R13—R15 do powrotu z procedury obsługi.

5. Maska przerwań w rejestrze ST jest zmniejszona o 1 w celu zablokowania przerwań o priorytecie równym obsługiwanemu. Zerowy stan maski oznacza akceptowanie tylko przerwań niemaskowalnych, wartość maksymalna (15) — wszystkie przerwania dozwolone.

Powrót z procedury obsługi przerwania odbywa się za pomocą rozkazu RTWP, który odtwarza stary kontekst według stanu rejestrów R13—R15.

Warto zwrócić uwagę na trzy cechy systemu przerwań mikroprocesora TMS9900:

— przerwań jest tyle, ile ich priorytetów pamiętanych w rejestrze ST (dokładnie — 16)
— adres pola wektorów przerwań jest stały, równy 0
— adres wektora konkretnego przerwania jest jednoznacznie określony przez jego priorytet (WEKTOR=PRIORYTET ×4).

KONFERENCJE

Mikrokomputery i banki

W dniach 15 i 16 września 1983 r. odbędzie się w Luksemburgu (Benelux) międzynarodowe seminarium „Mikrokomputery i banki". Organizatorem tej imprezy jest znany francuski instytut badawczy bankowości INSIG (Institut de Recherche Interbancaire), niezwykle aktywny na arenie międzynarodowej w dziedzinie rozwoju metod i technik informatycznych w działalności banków. Zapowiedziana konferencja jest kontynuacją organizowanego już od kilku lat cyklu międzynarodowych specjalistycznych konferencji naukowych poświęconych komputeryzacji banków. Poprzednie seminarium, które odbyło się w dniach 20—22 kwietnia br. w Paryżu, poświęcone było problemowi tworzenia, zarządzania i wykorzystywania baz danych zawierających informacje o klientach banków. Tematyka seminarium wrześniowego, organizowanego tym razem poza terytorium Francji, ukierunkowana została na zastosowanie mikrokomputerów. Świadczy to, że banki szybko nadążają za aktualnymi trendami rozwoju sprzętu komputerowego, co jednak oznacza konieczność dokonania istotnych zmian form organizacji wewnętrznej banków w wyniku pogłębiającej się decentralizacji systemów informatycznych.

Tematyka konferencji:

- wybór sprzętu i oprogramowania
- zmiany w koncepcji opracowań projektowych
- rola tradycyjnego działu przetwarzania danych
- trudności w adaptacji personelu do nowych form organizacji.

Współorganizatorem seminarium jest Towarzystwo Międzynarodowych Targów w Luksemburgu, dokąd można zwracać się o dalsze informacje: **Societe des Foires Internationales de Luxembourg, S.A., Boîte postale 110, L-2011 Luxembourg.**

JOHN EVANS
Europejski Instytut Związków Zawodowych
Bruksela

Raport dla Klubu Rzymskiego: Mikroelektronika i społeczeństwo. Na dobre i złe (2)

Wpływ komputeryzacji na miejsce pracy

Na początku tego rozdziału[1] wykazaliśmy, że — oprócz umiejętności, wprawy i kwalifikacji — walory danego zawodu określa jeszcze szereg dodatkowych czynników. Obejmują one kontrolę pracownika nad tempem i intensywnością własnej pracy, możliwości kontaktów z innymi ludźmi, stopień niezależności od nadzoru, fizyczne środowisko pracy oraz wpływ tej pracy na zdrowie pracownika i jego bezpieczeństwo. Nie można tych czynników całkowicie oddzielić od umiejętności i kwalifikacji, ponieważ wiadomo, że stanowiska pracy wymagające wysokich kwalifikacji związane są na ogół z lepszymi warunkami i odwrotnie. Zastosowanie mikroelektroniki może zaś spowodować poprawę warunków pracy niezależnie od jej wpływu na rodzaj wymaganych kwalifikacji.

Co do kwalifikacji, nie można wykluczyć nowych trudności. W niektórych najcięższych pracach, wykonywanych w wyjątkowo przykrych warunkach, człowieka zastąpi maszyna, zwiększą się natomiast stresy i napięcia wynikające z wzajemnego oddziaływania w układzie człowiek-maszyna na takich odcinkach, na których oddziaływanie to nie miało poprzednio miejsca.

INTENSYWNOŚĆ PRACY

Mikroelektronika prowadzi do zwiększenia wkładu kapitału w tzw. pracochłonnych gałęziach przemysłu i usługach, w których dotąd przeważał udział siły roboczej. Pojawi się przeto nacisk ekonomiczny, aby nowy, kapitałochłonny sprzęt wykorzystywać bardziej intensywnie, z czym wiąże się ryzyko pogorszenia warunków pracy.

Wprowadzenie przetwarzania tekstowego, jak wykazaliśmy poprzednio, może prowadzić do rozdzielenia maszynopisania od funkcji administracyjnych w pracy sekretarek. Wystąpi tu nie tylko polaryzacja kwalifikacji, ale może zwiększyć się intensywność pracy w zakresie maszynopisania. Po pierwsze — wzrost specjalizacji sam przez się zmniejsza urozmaicenie pracy i zwiększa możliwość nadzoru zwierzchników nad tempem pracy. Po drugie — zastosowanie procesorów tekstowych wywołuje wzrost intensywności samego procesu maszynopisania, w porównaniu z konwencjonalną maszyną do pisania. Zastosowanie procesora może zwiększyć fizyczną wydajność pracy maszynistki przez skrócenie czasu zużywanego na ewentualne przepisywanie, poprawienie i korektę tekstów oraz zajmowanie się kalką i papierem. Przy tej samej organizacji pracy, operator procesora pisze dany tekst dwukrotnie szybciej niż maszynistka używająca konwencjonalnej maszyny. Oznacza to, że możliwości kontaktów z innymi ludźmi w czasie pracy maleją, zaś niebezpieczeństwo izolacji społecznej i intensywność pracy — zwiększa się.

Jeszcze raz trzeba jednak podkreślić, że — spełniwszy określone warunki w mniej scentralizowanych systemach organizacyjnych — wprowadzenie przetwarzania tekstowego można wykorzystać do zmniejszenia ilości czasu przeznaczonego na maszynopisanie i w konsekwencji zwiększenia ilości czasu na inne zadania administracyjne. Efektem takiej zmiany byłoby wówczas zarówno podniesienie prestiżu oraz odpowiedzialności pracy sekretarki, jak i zmniejszenie intensywności tej pracy. O różnych wariantach organizacyjnych była już mowa w pierwszej części niniejszego rozdziału.

Niebezpieczeństwo zwiększenia intensywności pracy istnieje wśród tych zawodów z sektora usług, które poprzednio nie nadawały się do automatyzacji lub mechanizacji. Wspomniano już o wpływie projektowania wspomaganego komputerem (CAD) na pracowników wykonujących pracę ręczną wymagającą wysokich kwalifikacji w fabrykach, w których zautomatyzowano zarówno proces projektowania, jak i proces produkcyjny. Wprowadzenie projektowania wspomaganego komputerem wpływa na pracę kreślarza i konstruktora. Z jednej strony — komputer może przejąć takie rutynowe i mało twórcze funkcje konstruktora, jak kreślenie, spisywania elementów oraz specyfikacje dotyczące technik produkcyjnych i technologii. Przyspiesza to sam proces powstawania projektu, a jednocześnie eliminuje z pracy kreślarza czynności żmudne i powtarzalne. Jednak z drugiej strony komputer może współdziałać bezpośrednio z konstruktorem w taki sposób, aby można w sposób ciągły zmieniać projekt, wprowadzając do komputera nowe dane, ujawniane w trakcie pracy.

To wzajemne oddziaływanie człowieka z maszyną może prowadzić do przyspieszenia pracy, zmniejszenia jej elastyczności i zmienności oraz nasilenia stresu. Krytycy twierdzą, że wskutek upowszechnienia projektowania wspomaganego komputerem zwiększy się wyobcowanie, osłabnie poczucie zadowolenia z pracy projektowej, co będzie skutkiem podporządkowania operatora (czyli projektanta) komputerowi.

Pogorszenie warunków pracy z powodu trudności wynikających z wzajemnego oddziaływania człowieka i maszyny, nie musi koniecznie wynikać ze wzrostu tempa pracy; może ono również być skutkiem zmniejszenia tempa pracy. Przykładowo, urzędnik z działu zamówień, posługujący się bezpośrednim systemem informacji, musi często zwracać się do komputera o informacje konieczne do podjęcia decyzji i udzielenie odpowiedzi klientowi. Jeżeli z powodu przeciążenia systemu komputerowego odpowiedź nie przychodzi natychmiast, to praca tego urzędnika może stać się szalenie frustrująca. Problem będzie tu zasadniczo polegał na zapewnieniu zgodności szybkości operacji systemu komputerowego z wymaganą szybkością współdziałania człowieka z tym systemem. Wymaga to podjęcia odpowiednich decyzji już na etapie projektowania, zatem w grę wchodzi tu kwestia natury ekonomicznej lub organizacyjnej, nie zaś technicznej.

Chęć intensywniejszego wykorzystania urządzeń może prowadzić nie tylko do zwiększenia tempa pracy, ale do wprowadzenia lub rozwinięcia bardziej intensywnych form

[1] W poprzednim numerze INFORMATYKI zamieściliśmy pierwszą część pracy Johna Evansa stanowiącej rozdział Raportu — przyp. red.

pracy, np. pracy zmianowej. W jednym z europejskich zakładów przemysłowych, w którym zastosowano urządzenia elektroniczne do produkcji automatycznych maszyn kasowych, zostały wprowadzone dość szeroko systemy pracy zespołowej; jednak praca zmianowa praktykowana jest tam tylko na wydziale, na którym używa się robotów do montowania układów scalonych na płytkach drukowanych. Ze względu na wysoki koszt zastosowanych urządzeń, kierownictwo zakładu wprowadziło tam system pracy ciągłej. Wprowadzenie dużych systemów komputerowych w zakładach wiąże się z wprowadzaniem pracy zmianowej, a więc do pewnego stopnia także — pracy nocnej.

Praca zmianowa jest zdecydowanie niekorzystna pod względem społecznym i rodzinnym, a praca nocna ma szczególnie szkodliwy wpływ na zdrowie pracowników i higienę pracy. Dwa najpoważniejsze mankamenty to — uszczerbek snu nocnego i kłopoty trawienne. Poza tym, z licznych badań przeprowadzanych w szeregu krajów wynika, że większość pracowników nie lubi nocnej pracy i — o ile to tylko możliwe — stara się jej uniknąć.

IZOLACJA

Zarówno zwiększenie zakresu współdziałania w układzie człowiek-maszyna, jak i szersze występowanie pracy zmianowej dzięki mikroelektronice, zwiększyły zagrożenie izolacją człowieka w miejscu pracy. Przytoczone powyżej przykłady wskazują, jak wprowadzenie przetwarzania tekstowego do maszynopisania zwiększa izolację wskutek wzrostu intensywności pracy i natężenia hałasu. Podobnie, jak wykazano poprzednio, nie tylko maleją wymagania dotyczące kwalifikacji operatora komputerowych obrabiarek sterowanych numerycznie (CNC), ale także ulegają ograniczeniu jego bezpośrednie kontakty z innymi operatorami, konstruktorami i inżynierami. Obniżenie liczebności załogi w częściowo lub całkowicie zautomatyzowanych zakładach zwiększa ryzyko ograniczenia kontaktów społecznych podczas pracy. Dotyczy to w szczególności nocnej zmiany w zakładzie zautomatyzowanym, kiedy w pracy przebywa najmniej pracowników, a wszelkie prace konserwacyjne związane z obecnością innych ludzi w zakładzie przeprowadza się tam w dzień.

Ponieważ miejsce pracy jest głównym źródłem kontaktów społecznych w życiu ludzi, wzrost izolacji w pracy może wywołać większe stresy i wpłynąć ujemnie na stan zdrowia pracowników. Może też w znacznym stopniu obniżyć szansę kolektywnych działań w ramach związków zawodowych, skoro kontakty wśród załogi zostaną poważnie ograniczone. Badania związane z wprowadzeniem systemów komputerowych wykazują, że poziome kontakty między pracownikami zostają ograniczone, utrzymują się natomiast głównie hierarchiczne kontakty ze zwierzchnikami i podwładnymi.

Izolację może też potęgować przenoszenie pracy z zakładu do domu, co właśnie staje się możliwe dzięki mikroelektronice. Praca nakładcza odgrywała tradycyjnie ważną rolę w takich gałęziach, jak przemysł odzieżowy i włókienniczy. Dla pracodawcy główną zaletą pracy w domu są wynikające z tego tytułu oszczędności, nie ma bowiem potrzeby budować ani wyposażać fabryki; przeważnie też można mniej płacić pracownikom, gdyż domowe zatrudnienie zmniejsza możliwość podejmowania zbiorowych akcji związkowych. Ponadto, pracowników nakładczych można o wiele łatwiej zwolnić z pracy, niż — zatrudnionych w fabryce. Mikroelektronika stwarza możliwości techniczne do rozszerzenia zakresu pracy w domu w takich dziedzinach, jak np. praca biurowa. Terminale komputerowe można zainstalować w domach pracowników i dołączać je do komputera centralnego przez system telekomunikacyjny. Praca może więc być wykonywana w domu, ale jednocześnie — regulowana i nadzorowana w centrali.

Jeżeli dzięki mikroelektronice rozpowszechni się praca w domu i nastąpi likwidacja tradycyjnych stanowisk pracy, to mniej kobiet będzie miało możliwość „chodzenia do pracy". Wpłynie to na poważne zwiększenie ich izolacji i zahamuje postęp społeczny w kierunku równouprawnienia płci.

Wyjątkowo podatne na automatyzację przez zastosowanie mikroelektroniki są wszystkie prace polegające na powtarzaniu tych samych czynności lub ciągów czynności. W chwili obecnej są one wykonywane głównie przez kobiety, toteż musimy liczyć się z ryzykiem wzrostu bezrobocia właśnie wśród kobiet. Z powodu ograniczeń społecznych i organizacyjnych, kobietom trudniej przechodzić do nowych zawodów, wymagających znacznie wyższych kwalifikacji.

NADZÓR I KONTROLA

Zwróciliśmy już uwagę, że komputeryzacja może prowadzić do zmiany tempa pracy w wielu rodzajach zajęć, których poprzednio nie dotyczyły problemy wzajemnego oddziaływania w układzie człowiek-maszyna. Sam proces wprowadzania maszyn do dziedzin tradycyjnie pracochłonnych powoduje jednak zmniejszenie zakresu swobody człowieka w sposobie wykonywania pracy. Ilustrują to przytoczone poprzednio przykłady zmian charakteru pracy operatora obrabiarki i maszynistki. W obydwu przypadkach, wskutek komputeryzacji, maleje zakres samodzielności i słabną kontakty związane z podejmowaniem konkretnych decyzji podczas pracy. W rozdziale pt. „Techniki informacyjne a społeczeństwo" wskazano na niebezpieczeństwa zwiększonej kontroli nad pracownikiem w zakładzie pracy, wynikającej z zastosowania nowej technologii.

Mikroelektronika umożliwia ponadto rejestrowanie dokładnych informacji na temat pracy poszczególnych osób. Można to realizować różnymi metodami. Otóż przede wszystkim — można wykorzystać inteligentne terminale do gromadzenia różnych informacji, np. o czasie spędzonym przez pracownika przy maszynie, o szybkości jego pracy, o częstotliwościach popełniania omyłek i błędów itd. Wprowadzenie tego typu nadzoru prowadzi do zwiększenia stresów u pracownika oraz jednocześnie — do ściślejszej, odgórnej kontroli dotyczącej tempa pracy w zakładzie. Odnosi się to naturalnie do osób pracujących w sposób ciągły, do których zaliczamy m.in. kasjerki w domach towarowych, operatorów procesorów tekstowych lub obrabiarek sterowanych przez komputer. Szczególnie ostra dyskusja rozgorzała na ten temat w 1978 roku, w Danii, kiedy związek zawodowy pracowników handlu ujawnił, że wprowadzone wówczas do eksploatacji w sklepach kasy elektroniczne mają wbudowane urządzenia do rejestrowania informacji o pracy kasjerek i kasjerów.

Te ułatwienia w gromadzeniu informacji przez centralę stwarzają możliwość tworzenia komputerowych archiwów akt personalnych w instytucjach i zakładach pracy. Możliwość rejestrowania danych o sposobie wykonywania pracy, dyscyplinie i punktualności, w połączeniu z charakterystyką człowieka, dotyczącą np. jego działalności w organizacji związkowej, stanowi zagrożenie dla zasady nieingerencji w prywatne sfery życia człowieka. W rozdziale pt. „Techniki informacyjne a społeczeństwo" mówi się o sposobie, w jaki zmiany dotyczące tajemnicy osobistej i wolności człowieka łączą się z szerszą problematyką prywatności i swobody działania ludzi w skomputeryzowanym społeczeństwie.

MONITORY EKRANOWE I SKUTKI ERGONOMICZNE ICH UŻYCIA

Monitory ekranowe, inaczej ekranopisy (ang. visual display unit — VDU), zwane również lampami elektronopromieniowymi (cathode ray tube — CRT), wyposażone w klawiaturę, stały się standardowym środkiem komunikacji między ludźmi i urządzeniami komputerowymi. Wyparły dawniejsze techniczne środki komunikacji, jak np. karty i taśmy dziurkowane. Doprowadziło to do wyeliminowania szeregu zawodów wiążących się poprzednio z przygotowywaniem danych do komputerów. Wprowadzenie ekranopisów pociąga za sobą także pewne problemy dotyczące bezpieczeństwa i higieny pracy operatorów.

Zastosowanie monitorów ekranowych zwiększa prawdopodobieństwo przemęczenia wzroku, stresów, zmęczenia, bólu pleców i krzyża, bólów głowy i stanów zapalnych skóry. Z badań socjologicznych — przeprowadzonych przez organizacje związkowe na temat skutków wprowadzania ekranopisów w dziesięciu krajach — wynika, że problemy te mają raczej uniwersalny charakter. Szereg związków stwierdziło ponadto, iż monotonia i izolacja społeczna zwiększa się wraz z wprowadzeniem monitorów ekranowych i może wpływać szkodliwie na zdrowie i samopoczucie pracowników. Faktyczne skutki zależą od szeregu złożonych czynników, związanych zarówno z samym urządzeniem technicznym, jak i z otoczeniem miejsca pracy. Co do samego urządzenia — w grę wchodzą następujące

czynniki: luminacja ekranu, barwa obrazu, rozdzielczość zapisu znaków, ich rozmiar i zajmowana przestrzeń, szybkość migotania, implozyjność i eksplozywność lampy elektronowej, promieniowanie, wytwarzane ciepło, hałas, kształt obudowy, sposób konserwacji i rodzaj klawiatury. Czynniki zależne od miejsca pracy obejmują m.in. oświetlenie i oślepiające odblaski, ilość pracy papierkowej, ustawienie ekranu monitora, przestronność i zatłoczenie, pozycję przy pracy. Do istotnych czynników ergonomicznych można również zaliczyć regularne przerwy w pracy przy urządzeniach technicznych, umożliwiające kontaktowanie się z ludźmi i odpoczynek. Brak takich przerw oznaczałby, że operator ma spędzić całe swoje zawodowe życie wpatrując się w ekran i mając kontakt jedynie z ekranowym obrazem rzeczywistości, nie zaś z samą rzeczywistością. Mogłoby to prowadzić do bardzo złożonych problemów natury psychologicznej i społecznej.

Ze względu na wagę problemów, które pojawiają się wraz z wprowadzaniem monitorów ekranowych, związki zawodowe w wielu krajach Europy Zachodniej prowadzą badania na ten temat oraz starają się odpowiednio ukierunkować zasady obsługi i wykorzystanie tych urządzeń. Technologia może rozmaicie wpływać na sytuację człowieka w pracy. Problemy ergonomiczne, będące skutkiem zastosowania nowych technologii w danym procesie produkcyjnym, można ograniczyć, zaś korzyści z tego płynące — zmaksymalizować; pod warunkiem, że celem modernizacji będzie stworzenie zdrowego środowiska pracy, nie zaś jedynie chęć zwiększenia — za wszelką cenę — wydajności pracy.

Zastosowanie monitorów ekranowych spełnia generalne wymagania bhp, jeżeli jednocześnie spełnione będą następujące warunki:

- — odblaski od naturalnego bądź sztucznego oświetlenia nie zakłócają czytelności obrazu na ekranie,
- — zapis na ekranie jest wyraźny i łatwy do odczytania,
- — hałas powodowany przez pracę urządzeń jest wyciszony do minimum,
- — rodzaj klawiatury, kształt konsoli i krzesła operatora odpowiadają normom ergonomicznym, mającym na względzie jak najmniejsze zmęczenie pracownika,
- — urządzenia z monitorami ekranowymi znajdują się w pomieszczeniach umożliwiających kontakty z innymi osobami, lecz bez narażania pracowników na nadmierne zatłoczenie i ciasnotę. Należy ponadto przeprowadzać regularne kontrole wzroku operatorów oraz umożliwić im robienie regularnych przerw w pracy. W niektórych krajach prowadzi się również badania, mające na celu ulepszanie produkowanych tam monitorów ekranowych pod względem wymagań ergonomicznych, np. przez zmianę prędkości migotania lub przez zaprojektowanie nowych rodzajów wskaźników.

POLARYZACJA ZAWODOWA

Ujemne skutki rozwoju mikroelektroniki przejawiające się w tym, że jakość i kwalifikacje jednych zawodów podwyższają się, a innych obniżają, stanowią argument potwierdzający pogląd, że mikroelektronika może zostać wykorzystana do dalszego polaryzowania i rozdzielania różnych grup zawodowych w ramach rynków pracy krajów uprzemysłowionych. Proces dzielenia i polaryzacji zawodów nie jest zjawiskiem nowym, ani też nie wynika jedynie z wprowadzenia mikroelektroniki czy nawet czynników technologicznych w ogóle.

Analizy przeprowadzone w szeregu krajów uprzemysłowionych wykazują, że rynki pracy podzieliły się na obszary, które w sumie tworzą hierarchię kwalifikacji, wynagrodzeń, bezpieczeństwa, statusu i władzy. Kategorie zawodowe znajdujące się na szczycie hierarchii obejmują tzw. wolne zawody oraz wszelką kadrę kierowniczą, z tym, że łączą się one z kategoriami zawodów wymagających wysokich kwalifikacji i specjalizacji przemysłowych. Zawody te wymagają stosunkowo wysokiego wykształcenia i wyszkolenia, uzyskanego w szkołach i uczelniach bądź też — przez długoletnią praktykę zawodową. Zapewniają one względnie wysoki status, wysokie wynagrodzenie oraz na ogół wysoki stopień satysfakcji zawodowej, dotyczącej np. samodzielności w wykonywaniu pracy, większych możliwości awansu oraz — przynajmniej w początkowym stadium trwającej obecnie recesji — znaczne poczucie bezpieczeństwa.

Na niższych szczeblach hierarchii znajdują się zawody, które wymagają niepełnych kwalifikacji. Do tej kategorii zaliczamy głównie niższy personel biurowy i administracyjny. Większość zawodów wymagających niskich kwalifikacji mogą wykonywać osoby z wykształceniem podstawowym. Sektor ten odznacza się dość wysokim stopniem wewnętrznej ruchliwości. Stopień satysfakcji zawodowej nie jest tu na ogół wysoki, jakkolwiek bywa on różny w zależności od rodzaju wykonywanej pracy.

Najniżej w tej hierarchii znajdują się pracownicy niewykwalifikowani i niewyszkoleni. Wykonują oni zawody wymagające niewielkiego wykształcenia i przygotowania specjalistycznego. Wiąże się z tym na ogół niski status, niska płaca, ograniczone możliwości awansu oraz generalnie — niski poziom satysfakcji zawodowej. Praca podlega tu ściślejszej kontroli i ostrzejszemu nadzorowi, zwykle występuje też większa rotacja siły roboczej. Badania dotyczące stosunku do pracy wykazują, iż robotnicy niewykwalifikowani przywiązują największą wagę do takich cech pracy, jak poczucie bezpieczeństwa, warunki fizyczne pracy i zarobki.

W dyskusjach na temat podziału na grupy zawodowe w ramach rynku pracy, eksperci powołują się często na okres dziewiętnastowiecznej rewolucji przemysłowej w Europie Zachodniej. Uprzemysłowienie i produkcja masowa były skutkiem podziału pracy, rozbicia bardzo wielu funkcji zawodowych na proste czynności oraz mechanizacji opartej na maszynie parowej. Przeobrażenie technologiczne było niewątpliwie siłą pobudzającą, która stymulowała rewolucję przemysłową, wiązało się ono ściśle z nowymi formami organizacyjnymi i rozwojem gospodarczym.

Ekonomiczne uzasadnienie podziału pracy bodaj po raz pierwszy wyłożył Adam Smith w pracy pt. „Bogactwo narodów" (The Wealth of Nations) wydanej w 1776 roku. Następnie, pisał o tym szerzej Charles Babbage w książce pt. „O ekonomii maszyn i fabrykantach" (On the Economy of Machinery and Manufacturers), wydanej w 1832 roku, zaś logiczne wnioski wysnuł Frederick Taylor w swej pracy pt. „Naukowe zarządzanie" (Scientific Management) z 1911 r.

Korzyścią ekonomiczną rozwoju społeczeństwa przemysłowego, opartego na specjalizacji i innowacjach technologicznych, było stworzenie możliwości masowej konsumpcji i ogólne podniesienie stopy życiowej. Faktycznie, w niektórych krajach, na pewnych etapach, robotnicy przyjmowali zasady tayloryzmu w zamian za podwyższenie zarobków. Pociągało to jednak znacznie koszty społeczne, wynikające z podziału kategorii zawodowych wskutek wydzielenia mnóstwa standardowych, uproszczonych i powtarzalnych rodzajów zajęć. Toteż wykorzystanie mikroelektroniki do dalszej polaryzacji zawodów nie jest ani zjawiskiem nowym, ani bynajmniej nieuniknionym. Stanowi ono raczej odzwierciedlenie celów, których osiągnięcie ma zapewnić nowa technologia. Podsumowuje to amerykański informatyk, J. Weizenbaum, twierdząc, że „...komputer, z którego korzysta obecnie elita technologiczna, nie jest przyczyną niczego. Jest on raczej instrumentem, który służy racjonalizacji, podtrzymaniu i przedłużeniu trwania najbardziej konserwatywnych, wręcz reakcyjnych, elementów ideologicznych obecnego Zeitgeistu[2])".

Mimo wszystko wśród różnych możliwych zastosowań mikroelektroniki, można wybierać. Od tego wyboru będzie w ostateczności zależeć generalny wpływ nowej technologii na ludzi pracy.

Praktyczne doświadczenia związane z alternatywnym zastosowaniem systemów opartych na mikroelektronice dowodzą, że zmiany technologiczne stwarzają pozytywne możliwości optymalizacji innych czynników, nie tylko samej wydajności produkcji. Jeżeli dzięki mikroelektronice miałaby jedynie wzrosnąć wydajność pracy, to nowa technologia spowoduje polaryzację zawodów. Przykładowo — wprowadzenie obrabiarek sterowanych numerycznie za pomocą komputera (CNC) obniży poziom kwalifikacji zawodowych operatora, jeśli celem byłyby jedynie oszczędności siły roboczej, lecz może to również prowadzić do wzbogacenia kwalifikacji operatora, jeżeli wprowadzenie nowych maszyn będzie połączone z nową elastyczną formą organi-

---

[2]) Zeitgeist — niem. duch czasu — przyp. red.

zacji pracy. Podobnie, różne rozwiązania są możliwe w przypadku zastosowań procesorów tekstowych do pracy biurowej. Nową technologię można wykorzystać bądź do zwiększenia intensywności pracy, bądź do zmniejszenia ilości czasu, jaki sekretarka musi poświęcać na maszynopisanie.

TECHNOLOGICZNE ALTERNATYWY

Z badań na temat zachodnioeuropejskich doświadczeń w dziedzinie zmiany zakresu obowiązków w różnych zawodach, opublikowanych przez Europejski Instytut Związków Zawodowych wynika, że istnieją konkretne przykłady takich zmian, dzięki którym uległy poprawie warunki pracy. Przykładowo — wzbogacenie zawodu prowadzące do prawidłowej równowagi między funkcjami administracyjnymi i funkcjami w zakresie maszynopisania, może spowodować pozytywne skutki ergonomiczne. Jak stwierdziliśmy, regularne przerwy w obsłudze urządzeń z monitorem ekranowym (VDU) mają zasadnicze znaczenie dla zdrowia, natomiast dzięki rotacji zadań sekretarka będzie spędzać tylko pewną część swego czasu pracy przy ekranopisie. Większość zachodnioeuropejskich ekspertów związkowych kładzie silny nacisk na to, aby zakres obowiązków sekretarki zawierał zastrzeżenie, że może ona spędzać jedynie część pracy przy procesorze tekstowym. Znalazło to już wyraz w szeregu układów zbiorowych dotyczących tej grupy zawodowej w niektórych krajach. Układ zbiorowy przyjęty w RFN w lipcu 1980 zawiera zastrzeżenie, że operatorzy monitorów ekranowych w sektorze bankowym nie powinni pracować przy maszynie dłużej niż cztery godziny dziennie. Dokument ten gwarantuje im również prawo do cogodzinnych przerw w obsłudze monitora oraz regularne kontrole lekarskie. Jeżeli rotacji zadań w ramach obowiązków służbowych będzie towarzyszyć przestrzeganie norm ergonomicznych, o których była mowa w poprzedniej części niniejszego rozdziału, to niebezpieczeństwa związane z pracą przy monitorze ekranowym mogą zostać znacznie zmniejszone.

Zastosowanie mikroelektroniki w procesach produkcyjnych i w wyrobach stanowi również bodziec do zdecentralizowania organizacji pracy i tworzenia małych, niezależnych jednostek produkcyjnych, podlegających jedynie centralnej koordynacji. Takie formy organizacji pracy są ułatwione dzięki zastosowaniu w procesie produkcyjnym systemów komputerowych rozłożonych przestrzennie, jednocześnie połączonych z komputerem centralnym. Jak już stwierdziliśmy, wprowadzenie nowych technologii może prowadzić do rozbicia linii produkcyjnej w zakładzie na mniejsze, niezależne jednostki.

Zastosowanie mikroelektroniki w wyrobach stanowi również potencjalny bodziec zachęcający do zmian w organizacji pracy. W latach siedemdziesiątych, w jednym z włoskich koncernów produkujących sprzęt biurowy, przeprowadzono szereg reform w systemie organizacji pracy. Były one skutkiem zarówno nacisku związków zawodowych, domagających się zmniejszenia monotonii pracy w pewnych zawodach, jak i starań kierownictwa dążącego do osiągnięcia większej elastyczności w systemie produkcji, z uwagi na przestawienie się zakładów z produkcji urządzeń mechanicznych na produkcję urządzeń elektronicznych.

Możliwości decentralizacji systemów produkcji, jakie stwarza mikroelektronika, mogą jednak poprawić albo pogorszyć warunki pracy. Jak uzasadniliśmy uprzednio, wykonywanie pracy zawodowej w większym zakresie w domu może prowadzić do większej izolacji społecznej pracownika. Tworzenie autonomicznych zespołów w rodzaju brygad pracy może również zostać wykorzystane do podważenia roli organizacji związkowych i organizacyjnego rozbicia świata pracy, jeżeli jednocześnie nie zostaną zapewnione odpowiednie warunki dla podtrzymania kontaktów społecznych i spotkań między różnymi zespołami ludzi pracujących razem. Tak więc wpływ, jaki wywrze technologia, będzie stanowić raczej odbicie społecznego wyboru co do sposobu jej użycia, nie zaś jakiś imperatyw natury technicznej.

Dość istotnym problemem związanym z technologią opartą na mikroelektronice jest fakt, iż można bazując na niezrozumiałych dla niespecjalistów szczegółach technicznych — dokonać arbitralnego wyboru co do jej zastosowania, ukrywając przed zainteresowanymi faktyczne czynniki społeczne i skutki ekonomiczne. Co więcej, oparte na mikroelektronice urządzenia komputerowe wprowadza się zwykle etapami i w chwili oddania pojedynczej maszyny do eksploatacji w zakładzie, dość trudny do przewidzenia jest efekt końcowy całego programu modernizacji. A przecież, wskutek połączenia pojedynczych części wyposażenia, efekt końcowy może znacznie przewyższać efekty, jakie powoduje zastosowanie poszczególnych części oddzielnie. Dla przykładu — wprowadzenie elektronicznych terminali kasowych w sklepach, co na pierwszy rzut oka nie wydaje się wpływać zbyt drastycznie na pracę personelu, może stanowić zaledwie część wielkiego programu komputeryzacji całej firmy, polegającego na komputeryzacji magazynowania towarów, kodowania ich cen na opakowaniu (specjalnym systemem pasków różnej grubości) oraz kontroli zapasów — a to wszystko razem pociąga za sobą znacznie większe implikacje dla wszystkich osób zatrudnionych w danej firmie handlowej.

IMPLIKACJE DLA STOSUNKÓW W PRZEMYŚLE

Doświadczenia wskazują zatem, że z mikroelektroniki można uczynić dwojaki użytek i wykorzystać ją do stworzenia mniej spolaryzowanej struktury społeczeństwa przemysłowego, bądź w celu zwiększenia centralizacji i polaryzacji. O wyborze decydują względy społeczne i polityczne, nie zaś technologiczne.

Z niniejszego rozdziału dowiedzieliśmy się już, że jeśli zmiana struktury pewnych zawodów nastąpi w kierunku zwiększania odpowiedzialności, urozmaicenia zakresu obowiązków, dania większej samodzielności pracownikowi — to w takim przypadku mikroelektronika posłuży poprawie warunków pracy i podwyższeniu kwalifikacji zawodowych. Zwróciliśmy uwagę, że bardzo istotne jako czynnik określający charakter pracy jest zastrzeżenie, by pracownik spędzał przy obsłudze terminala komputera jedynie pewną, ograniczoną część swego czasu pracy. Powinno się pamiętać o tym przy planowaniu podziału obowiązków w ramach nowych zawodów, które powstają wskutek wprowadzania nowej technologii.

Z dążeniem do poprawy warunków pracy może łączyć się konflikt interesów załogi i kierownictwa. W pewnych sytuacjach wszelako korzyści długofalowe powinny przesądzać o przyspieszeniu porozumienia przez negocjacje i kompromisy. Na tym przecież polega logika stosunków w przemyśle, opartych na zbiorowych przetargach. Zmiana przez porozumienie — to jedyna możliwa do przyjęcia dalekowzroczna zasada wszelkich efektywnych zmian w zarządzaniu.

Osiągnięcie w dłuższym okresie optymalnej równowagi pomiędzy maksymalizacją docelowych korzyści społecznych i ekonomicznych wymaga uczestnictwa wszystkich, których dotyczy proces zmian. Dla ludzi pracy, będących głównymi, bezpośrednimi użytkownikami nowej technologii, z uczestnictwem tym będą związane poważne zmiany dla ich własnej organizacji przedstawicielskiej, czyli dla związków zawodowych.

W okresach różnych przeobrażeń, na etapie, kiedy nie było wpływu na decyzje, związki zawodowe często przyjmowały postawę obronną w stosunku do zachodzących zmian, w zamiarze obrony interesów swych członkow. Wprawdzie działanie takie jest chwalebne, zwłaszcza gdy wszystkie inne środki zawiodą, ale kojarzy się z gaszeniem pożaru, który już wybuchł. Jeżeli nie chcemy zmarnować szansy, jakie stwarza nowa technologia, związki zawodowe muszą być informowane o planowanych zmianach na takim etapie procesu przeobrażeń, kiedy rzeczywiście mogłyby jeszcze wziąć udział w ocenie technicznych i społecznych względów decydujących o wyborze. Związki zawodowe powinny aktywnie uczestniczyć w podejmowaniu decyzji, nie zaś występować jedynie w roli czynnika zgłaszającego protest po fakcie, a więc — muszą działać na zasadzie akcji, nie tylko reakcji.

W wielu krajach jest to zupełnie nowa sytuacja, wymagająca wprowadzenia zmian w szeregu tradycyjnych sferach działania związków zawodowych. W niektórych krajach — na przykład — trzeba było sprowadzić przetargi i negocjacje na płaszczyznę korporacji (przedsiębiorstwa). W miarę, jak nowa technologia zmienia charakter poszczególnych zawodów i prowadzi do przechodzenia różnych grup zawodowych z jednego sektora do drugiego, granice między związkami zawodowymi, skupiającymi pracowników prze-

mysłu i rzemiosła, zacierają się, co pociąga za sobą konieczność zmian w strukturze organizacji związkowych. Coraz większego znaczenia nabiera też bliski kontakt między działaczami związkowymi uczestniczącymi bezpośrednio w negocjacjach, a reprezentowanymi przez nich grupami zawodowymi. Poza tym, dzięki mikroelektronice, większego znaczenia nabiera poziom wykształcenia działaczy związkowych, ich wyszkolenie i kontakt ze specjalistami z kręgów akademickich oraz z innymi środowiskami, których struktura związkowa bezpośrednio nie obejmuje.

Implikacje wypływające z wprowadzenia mikroelektroniki dla dyrekcji i generalnie — dla kierownictwa przedsiębiorstw omówiono w rozdziale pt. „Wpływ na przedsiębiorstwo". Pewne tradycyjne postawy wobec prerogatyw kierownictwa oraz zakres przetargów i negocjacji z udziałem załogi trzeba będzie dostosować do nowej sytuacji. Jeżeli społeczeństwa mają stawać się coraz bardziej demokratyczne, to nie do pomyślenia byłoby, aby wysoka stopa życiowa, świadomość polityczna i w ogóle — demokratyzacja w życiu społecznym, mogły współistnieć z coraz bardziej spolaryzowaną strukturą pracy, opartą na zasadach tayloryzmu.

Szereg krajów przemysłowych podjęło już działania na tym polu. W Norwegii — na przykład — podpisano w 1975 roku ogólnokrajowe porozumienie między organizacją związkową i organizacjami skupiającymi pracodawców. Porozumienie dotyczy zasad wdrażania systemów opartych na komputerach. Zaleca ono jak najwcześniejsze informowanie związków przez kierownictwo o projektowanych innowacjach, jeszcze przed podjęciem wiążących decyzji i zakłada, że zarówno dyrekcja, jak i związki powinny w możliwie najszerszym zakresie uświadamiać załodze planowane zmiany. Załogi mają prawo wybierać fachowych przedstawicieli, którzy będą wyjaśniać kwestie technologiczne. Zapewnia się im również przeszkolenie w specjalnościach związanych z systemami komputerowymi; zaczęto ich nawet nazywać „dostawcami danych" (ang. data stewards).

W szeregu innych krajów podpisano istotne porozumienia dotyczące poszczególnych sektorów, zwłaszcza tych, które najbardziej bezpośrednio odczuwają wpływ mikroelektroniki. W większości państw Europy Zachodniej zawarto np. porozumienie dotyczące przemysłu poligraficznego, w związku z wprowadzeniem skomputeryzowanego składu. Dokumenty te zwykle uwzględniają takie sprawy, jak bezpieczeństwo i higiena pracy, liczebność załogi, zasady szkolenia w związku ze zmianą kwalifikacji oraz zmiany stanowisk, granicę obniżania kategorii zawodowych oraz poziomu uposażenia pracowników przeniesionych na inne stanowiska w związku z modernizacją. W niektórych krajach Europy Zachodniej podpisano również porozumienia dotyczące sektora usług pocztowych i telekomunikacji, gdzie także znalazła zastosowanie nowa technologia (np. w Wielkiej Brytanii i Holandii), oraz w przemyśle metalowym (np. w części Republiki Federalnej Niemiec i we Włoszech).

Niezależnie od porozumień w skali ogólnokrajowej lub w skali poszczególnych sektorów, chyba najistotniejsze decyzje w sprawie zastosowań mikroelektroniki oraz strategii planowania zapadają na pozomie korporacji lub przedsiębiorstwa. W szeregu krajów związki zawodowe w przedsiębiorstwach zorganizowały komórki zajmujące się negocjacjami w sprawach technologicznych. W Republice Federalnej Niemiec rady zakładowe stały się jednym z głównych instrumentów, za pośrednictwem których organizacje związkowe uzyskują informacje i wywierają wpływ na decyzje dotyczące wdrażania nowej technologii na poziomie korporacji. Rady te mają prawne gwarancje współdecydowania, konsultacji i uzyskiwania informacji na temat ogólnych problemów polityki przedsiębiorstwa, dotyczących zmian technologii, nie mają jedynie prawa udziału w negocjowaniu ostatecznych układów zbiorowych.

Szereg porozumień dotyczących wprowadzania technologii na poziomie korporacji podpisano w Wielkiej Brytanii. Dokumenty te dotyczą głównie konsultacji i wczesnego zawiadamiania o planowanym wprowadzeniu nowej technologii oraz określają na nowo stosunek istniejącej machiny związkowej i obowiązującej procedury negocjacyjnej do struktur i zasad działania związków w odniesieniu do tej technologii. W porozumieniach uwzględniono specyfikę bhp, granice obniżania kategorii zawodowych oraz wymagania dotyczące zmiany kwalifikacji w związku z wprowadzaniem nowej technologii. Zwykle dokumenty takie zawierają również wyszczególnienie norm ergonomicznych, przewidzianych dla osób obsługujących terminale komputerowe i monitory ekranowe, obejmują też wykaz zakresu obowiązków na danym stanowisku, łącznie z określeniem zasad organizacji pracy. Ostatnia sprawa, którą poruszaszają układy zbiorowe, dotyczy wpływu nowej technologii na czas pracy i wynagrodzenie pracowników.

Dzięki rozwojowi negocjacji w sprawie wprowadzania nowej technologii uświadomiono sobie znaczenie ściślejszych więzi między obieralnymi przedstawicielami związkowymi i masami członkowskimi. Nabrał też wagi problem stworzenia pełnego programu edukacji mas związkowych w dziedzinie nowej technologii. W Norwegii — na przykład — już od początku lat siedemdziesiątych krajowe centrum obliczeniowe i związki zawodowe przemysłu metalowego wspólnie organizują kursy dokształcające dla członków związku. Wydano dwa podręczniki dla związkowców na temat przetwarzania danych, planowania oraz sterowania i kontroli produkcji. Tę współpracę naukowców ze związkami zawodowymi podjęto następnie w innych sektorach, a także w innych krajach — zwłaszcza w Danii, RFN i Szwecji. Jednym z najistotniejszych motywów tych doświadczeń była konieczność zachęcenia szeregowych mas związkowych do przeanalizowania własnych celów ze zmianą technologiczną.

Związki zawodowe w Szwecji zaczęły korzystać z usług konsultantów doradzających jak podnieść zarobki. W niektórych przypadkach związki przeprowadzają pewne operacje przy pomocy konsultantów wyspecjalizowanych w nowych systemach produkcji, którzy pozostają zresztą na liście płac pracodawcy w danym przedsiębiorstwie. Związki zawodowe zdały sobie również sprawę z tego, jak ogromne znaczenie ma dostęp do środków przeznaczonych na działalność badawczo-rozwojową (B+R), wskutek czego podjęto starania zmierzające do zwiększenia wpływu na przydział tej części krajowych funduszy z budżetu B+R, którą przeznacza się na cele B+R w przemyśle.

Systemy powszechnej oświaty w państwach uprzemysłowionych będzie trzeba również dostosować do zmienionych warunków. Ich zadaniem będzie nie tylko dostarczanie liczniejszej kadry specjalistów komputerowych, których coraz więcej będą potrzebowali producenci komputerów. Do zadań szkolnictwa będzie należało również wykształcenie licznej kadry nowych specjalistów, przygotowanych do pracy z udziałem komputera. Oznacza to odmistyfikowanie wiedzy komputerowej w systemie szkolnictwa powszechnego. Równocześnie należy uczulić specjalistów na społeczne i organizacyjne konsekwencje upowszechnienia się systemów komputerowych.

I wreszcie — rządy powinny podjąć działania na rzecz stworzenia środowiska makroekonomicznego, w którym można skutecznie przeprowadzać zmiany technologiczne. Wspólne negocjowania układów zbiorowych tylko wówczas wyeliminuje ujemne skutki zmiany technologicznej, jeżeli będzie odbywać się w warunkach generalnego wzrostu zatrudnienia, stopy życiowej i zaspokojenia aspiracji społecznych ludzi pracy. O tych problemach traktuje następny rozdział, zatytułowany „Mikroelektronika a makroekonomia". Naszym zadaniem na lata osiemdziesiąte winno być zatem skuteczne rozwiązanie kwestii bezrobocia, przez skupienie się na zagadnieniach obejmujących skrócenie czasu pracy, skoordynowany program działania na rzecz odnowy gospodarczej i ukierunkowanie rozwoju na tworzenia nowych miejsc pracy. Jeżeli tego nie uczynimy, potencjał, jaki oferuje nam nowa technologia, zostanie zmarnowany wskutek wzrostu bezrobocia. I na tym tle robotnicy będą walczyć o poczucie bezpieczeństwa, występując **przeciwko** zmianom, skoro one same im takiego poczucia nie zagwarantują.

Tłumaczyła EWA WOYDYŁŁO

WACŁAW ISZKOWSKI, MAREK MANIECKI
Instytut Informatyki
Politechnika Warszawska

Standaryzacja języka BASIC (2)

Moduł rdzeniowy języka

W poprzednim numerze INFORMATYKI omówiliśmy normę minimalnej wersji języka BASIC, zatwierdzonej przez międzynarodowe organizacje normalizacyjne (ISO, ECMA, i ANSI). Od kilku lat trwają — i wydają się być już na ukończeniu — prace nad normą pełnego języka [1, 2]. Jej zakres obejmuje moduł rdzeniowy, omawiany w tej części artykułu, oraz pięć modułów rozszerzających cechy funkcjonalne języka, które przedstawimy w części następnej.

Omówiona poprzednio minimalna wersja języka BASIC jest w zasadzie podzbiorem modułu rdzeniowego. Jedyną istotną różnicą, mającą wpływ na przenośność programów napisanych w języku minimalnym na instalację z implementacją wersji rdzeniowej, jest wymaganie, aby nie używać deklaracji standardowej tablic, które muszą być wcześniej zadeklarowane instrukcją deklaracji DIM. Wymaganie to wynika z możliwości wykorzystywania w pełnej wersji języka dowolnych identyfikatorów zmiennych.

Rozszerzenie języka obejmuje m.in. zwiększenie dopuszczalnej długości sekwencji znaków w napisach, wraz ze wzbogaceniem repertuaru operacji napisowych. Wprowadzono również znacznie efektywniejsze instrukcje sterujące i iteracyjne, umożliwiając stosowanie zasad programowania strukturalnego w zapisie programów. Włączono też możliwość podziału programu na rozłączne jednostki oraz programową obsługę protestów.

Poniżej omówimy główne rozszerzenia zawarte w propozycji normy modułu rdzeniowego w stosunku do normy języka minimalnego. W dalszym ciągu zakładamy, że większość czytelników zna podstawowe cechy języka BASIC lub jedną z jego realizacji.

TEKST PROGRAMU

Program w języku BASIC składa się z pewnej liczby jednostek programowych. Pierwszą z nich jest program główny, po którym mogą występować definicje podprogramów, funkcji zewnętrznych itp. Program może być identyfikowany linią instrukcji PROGRAM, opisującą jego parametry. Instrukcja END kończy program główny oraz pozostałe jednostki programowe.

Linie tekstu programu mogą zawierać do 132 znaków z możliwością podziału linii na kolejne wiersze połączone znakiem &. W języku dopuszcza się pełny zbiór znaków kodu ASCII, z małymi literami włącznie, przy czym znaczenie małych i dużych liter poza napisami jest tożsame.

Instrukcję komentarza REM można zastąpić instrukcją pustą identyfikowaną znakiem wykrzyknika, po którym może być umieszczony tekst komentarza. Analogicznie, komentarzem można uzupełnić każdą linię.

Istotna jest możliwość użycia dowolnych identyfikatorów (o długości do 31 znaków) rozpoczynających się od litery i złożonych z liter, cyfr oraz znaku podkreślenia. Identyfikatory zmiennych napisowych są kończone znakiem dolara $. Używanie niektórych identyfikatorów będących słowami kluczowymi instrukcji i funkcji zostało zastrzeżone.

OPERACJE NUMERYCZNE

W rozszerzonej wersji języka dopuszcza się operacje numeryczne nadal jedynie na liczbach typu zmiennoprzecinkowego dziesiętnego. Ich reprezentacja wewnętrzna musi uwzględniać precyzję obliczeń z dokładnością co najmniej do m+1 cyfr dziesiętnych (m jest precyzją zapisu). Dodatkowo zezwala się na wprowadzenie rodzimej reprezentacji wewnętrznej wartości numerycznych. Określenia konwencji operacji numerycznych dokonuje się instrukcją OPTION ARITHMETIC. Podobnie określa się konwencję wartości kątowych (w radianach lub stopniach), wykorzystywanych jako parametry lub wartości funkcji trygonometrycznych.

Zmienne numeryczne proste o dowolnych identyfikatorach, deklarowane implicite w jednostce programu, są w niej lokalne. Wartości początkowe tych zmiennych są zależne od implementacji.

W odróżnieniu od normy języka minimalnego wszystkie zmienne tablice muszą być przed wykorzystaniem deklarowane instrukcją DIM lub DECLARE NUMERIC. Nowością tej wersji języka jest możliwość określania dolnej granicy zakresu indeksów. Przykładowo linia:

```
1010 DIM lata (1967 TO 1982)
```

deklaruje wektor lata obejmujący roczniki od 1967 do 1982. Przy takim rozszerzeniu wersji minimalnej instrukcja OPTION BASE pozostała jedynie dla zachowania równoważności obu wersji języka. Jest prawdopodobne, że w następnym wydaniu normy ta opcja zostanie usunięta.

W celu zapewnienia możliwości programowego stwierdzania zakresów tablic i ich rozmiarów, zdefiniowano dodatkowe trzy funkcje, dostarczające takich informacji (SIZE, LBOUND i UBOUND).

Składnia i semantyka wyrażeń numerycznych pozostaje bez zmian w stosunku do wersji minimalnej. Rozszerzeniem instrukcji podstawienia (słowo kluczowe LET jest nadal obowiązkowe) jest możliwość równoczesnego nadania wartości początkowych kilku zmiennym wymienionym na liście podstawienia. Przykładowo można zapisać instrukcję:

```
1020 LET rok, lata (1968), lata (1970), x = 1.5
```

nadającą wartość 1.5 wymienionym zmiennym prostym rok i x oraz dwóm elementom tablicy lata.

FUNKCJE NUMERYCZNE

W rozszerzonej wersji języka zdefiniowano ogółem 38 standardowych funkcji numerycznych, bezparametrowych oraz jedno- lub dwuparametrowych. Ponadto, funkcje numeryczne mogą być zdefiniowane jako wewnętrzne lub zewnętrzne jednostki programu. Definicja funkcji wewnętrznej (DEF) może być wieloparametrowa, złożona z wielu linii i zakończona instrukcją końca definicji END DEF. W definicji funkcji musi wystąpić co najmniej jedna instrukcja podstawienia LET, przypisująca wartość identyfikatorowi funkcji. Do zakończenia wykonywania funkcji, oprócz instrukcji END DEF, służy instrukcja wyjścia EXIT DEF. Identyfikator funkcji może być dowolny (w tym również FNx).

Funkcję FNx zdefiniowaną uprzednio możemy więc zapisać również jako:

```
50 DEF fnX (parametr)
51 LET wart=parametr * parametr
52 LET pom=(wart + 1)/(wart—1)
53 LET fnx=pom+Y
55 END DEF
```

Definicje funkcji zewnętrznych są traktowane jako osobne jednostki programu. W jednostce wykorzystującej je muszą być zadeklarowane przed użyciem instrukcją deklaracji funkcji (FUNCTION).

INSTRUKCJE STERUJĄCE

Repertuar instrukcji sterujących jest identyczny jak dla języka minimalnego, jednak efektywność ich działania jest znacznie większa.

Instrukcją przełączania ON można dokonywać przejść do numerów linii wybranych według wartości wyrażenia oraz przejść ze śladem do linii początków podprogramów. Jednocześnie, w obu przypadkach istnieje możliwość uzupełnienia instrukcji zwrotem: ELSE instrukcja wykonywana w przypadku, gdy dla danej wartości wyrażenia nie przewidziano odpowiedniego przejścia.

Instrukcja warunkowa IF jest w tej wersji języka efektywną instrukcją decyzyjną. Przede wszystkim uzupełniono ją zwrotem:

ELSE numer_linii

oraz dopuszczono bezpośrednie ustawianie instrukcji zamiast numerów linii po słowach THEN i EISE, co w wielu przypadkach skraca zapis algorytmu. Inne rozszerzenie definiuje strukturalną instrukcję warunkową, rozłożoną na rozłączne linie, umożliwiającą zapisanie wielu linii instrukcji dla różnych alternatyw. Dlatego pojawiła się linia instrukcji ELSE oraz:

ELSIF wyrażenia_relacyjne THEN

Końcem takiej instrukcji warunku jest linia END IF.

Składnia wyrażenia relacyjnego, występującego w instrukcji warunkowej, jest uzupełniona o operatory logiczne, dla których argumentami są relacje. Zasady obliczania wartości wyrażeń relacyjnych są następujące:

- wartość wyrażenia oblicza się od strony lewej do prawej z uwzględnieniem nawiasów
- po obliczeniu wartości obu argumentów (dla operatora NOT — jednego) operacji logicznej obliczana jest jej wartość
- jeżeli wyrażenie jest dysjunkcją, to pierwsza obliczona wartość logiczna równa prawdzie jest wartością wyrażenia bez obliczania jego pozostałej części
- jeżeli wyrażenie jest koniunkcją, to pierwsza obliczona wartość logiczna równa fałszowi jest wartością wyrażenia.

Przykładowo — dla wyrażenia.

X>0 OR (B=2 AND X=0)

oraz X równego 2 relacja X>0 jest prawdziwa, a więc obliczanie wartości wyrażenia zostanie na tym zakończone. Z kolei dla X równego —3 oraz B równego 1, po obliczeniu wartości relacji X>0, obliczana będzie wartość wyrażenia w nawiasie. Ponieważ relacja B=2 jest fałszywa, całe wyrażenie w nawiasie jest fałszywe i nie ma potrzeby obliczania relacji X=0.

W celu zilustrowania postaci instrukcji warunkowej zapiszemy w kilku wariantach przykład podany dla języka minimalnego.

a)

```
60 IF X>1 THEN 90 ELSE 70
70    LET  Y = X
80    GOTO  100
90 LET Y=LOG(X)
100 REM
```

b)

```
60 IF X>1 THEN LET Y=LOG(X)ELSE Y=X
```

c)

```
60 IF X>1 THEN
70     LET  Y = LOG(X)
80 ELSE
90     LET  Y = X
100 END IF
```

Dodatkową instrukcją decyzyjną jest instrukcja strukturalna wyboru. Pierwszą jej linią jest instrukcja selekcji:

SELECT CASE wyrażenie

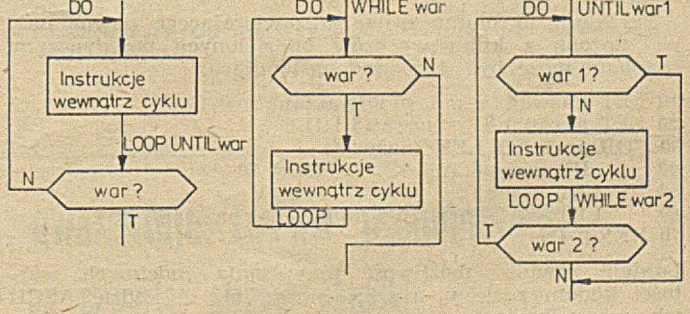
Poszczególne przypadki selekcji są oznaczone instrukcjami: CASE zakres lub CASE ELSE ... (wszystkie pozostałe) natomiast koniec instrukcji wyboru jest identyfikowany instrukcją END CASE. Zamiast dokładnego opisu jej działania podamy przykład użycia:

```
100 SELECT CASE X
110 CASE IS<0
120     PRINT    "test ujemne"
130 CASE 0
140     PRINT X; "równe zero"
150 CASE 2 TO 5
160     PRINT X; "należy do przedziału A"
170 CASE ELSE
180     PRINT X; "pozostałe wartości"
190 END CASE
```

INSTRUKCJE ITERACYJNE

Instrukcja cyklu wyliczanego jest identyczna jak dla języka mininmalego. Dodatkowo wprowadzono nową klasę instrukcji iteracyjnych — instrukcje cyklu warunkowego. Ich działanie jest oparte na instrukcji cyklu nieskończonego DO..LOOP, wykonującej cyklicznie instrukcje zawarte między linią rozpoczęcia cyklu DO a linią końca cyklu LOOP.
Zdefiniowano zwroty sprawdzania warunku:

WHILE wyrażenie_relacyjne
oraz
UNTIL wyrażenie_relacyjne




```
DO                  DO WHILE war        DO UNTIL war1
...                 ...                 ...
LOOP UNITL war      LOOP                LOOP WHILE war2
```

Schematy instrukcji cyklu warunkowego

Każdy z nich powoduje zaprzestanie wykonywania cyklu w przypadku, gdy wartość wyrażenia relacyjnego jest fałszem (dla zwrotu WHILE) lub prawdą (dla zwrotu UNTIL). Uzupełniając tymi zwrotami instrukcję rozpoczęcia cyklu DO, instrukcję końca cyklu LOOP lub też obie te instrukcje, otrzymamy różne postacie instrukcji cyklu warunkowego (rys.).

Dodatkowo, dla zmiany sekwencji wykonywanych instrukcji zdefiniowano instrukcję wyjścia z wnętrza instrukcji strukturalnych cyklu wykonanego (EXIT FOR) oraz cyklu nieskończonego lub warunkowego (EXIT DO). Efektem działania tej instrukcji jest przejście do następnej linii po instrukcji strukturalnej.

OPERACJE NAPISOWE

Sekwencja znaków może być dowolnej długości, ograniczonej dla danej implementacji lub w deklaracji zmiennych typu napisowego. Długość stałej napisowej zależy więc tylko od implementacji.

Identyfikatory napisów różnią się od pozostałych znakiem dolara $. Zmienne proste napisowe nie muszą być wcześniej deklarowane. Dopuszczalne jest używanie tablic napisowych jedno- lub dwuwymiarowych.

Z każdą zmienną napisową prostą lub złożoną może być związany kwalifikator określający wycinek napisu będącego wartością tej zmiennej (poprzez podanie numeru pierwszego i ostatniego znaku wycinka w napisie). Zmienna z kwalifikatorem wycinka napisu może być argumentem wyrażenia napisowego lub zmienną podstawianą. Przykładowo — w instrukcjach podstawienia:

```
10 LET nazwa$ = „PROGRAMOWANIE"
20 LET miara $ =nazwa $ (4:7)
```

otrzymujemy wartość zmiennej miara $ równą „GRAM" (element (4:7) jest kwalifikatorem wycinka).

Argumentem wyrażenia napisowego może być stała lub zmienna napisowa oraz funkcja napisowa, standardowa lub zdefiniowana wewnętrznie bądź zewnętrznie. W języku zdefiniowano osiem napisowych funkcji standardowych o wartościach typu napisowego oraz cztery funkcje napisowe o wartościach numerycznych. Funkcje te umożliwiają proste przekształcanie zawartości napisu.

Jedynym operatorem wyrażenia napisowego jest operator konkatenacji łączący dwa argumenty typu napisowego w jeden napis. Również relacje w wyrażeniach relacyjnych mogą dotyczyć porównywania zgodności wyrażeń napisowych. W rozszerzeniu języka uniwersalnego zdefiniowano też relacje większości i mniejszości napisów, ze względu na kolejność uporządkowania znaków (np. alfabetyczną — dla liter).

W języku wprowadzono dodatkowo deklarację zmiennych napisowych, z możliwością specyfikacji maksymalnej długości napisu. Przykładowo — linia:

```
100 DECLARE STRING alfa$ * 8, line$ * 10
```

jest deklaracją zmiennej alfa$ o maksymalnej długości napisu równej 8 i zmiennej line$ o długości 10.

Uzyskanie akronimu z napisu zawierającego pewną nazwę złożoną z kilku wyrazów oddzielonych pojedynczymi spacjami może być dokonane następująco:

```
10 LET nazwa$ =... ! przetwarzana nazwa
20 LET akronim$ = nazwa$(1:1)
30 FOR i=2 TO LEN (nazwa$)
40     IF nazwa$ (i,i) <>" " THEN 70
50     LET i=i+1
60     LET akronim$ = arkonim$ & nazwa$(i:i)
70 NEXT i
```

Istnieje również możliwość przypisania rodzimych wartości kodom znaków, a więc innego niż w kodzie ASCII ich uporządkowania.

MODULARNOŚĆ PROGRAMU

W rozszerzonej wersji języka wprowadzono (nareszcie!) możliwość podziału programu na jednostki będące:

- podprogramami
- funkcjami zewnętrznymi
- innymi programami.

Wszystkie identyfikatory występujące w jednostce są w niej lokalne. Wiązanie parametrów z argumentami następuje przy użyciu instrukcji wywołania podprogramu (CALL), podczas wywołania funkcji oraz wykonania instrukcji wywołania programu (CHAIN). Wartości argumentów są kopiowane na parametry lokalne jednostki, a dla podprogramu — kopiowane powtórnie po jego zakończeniu.

Definicja funkcji zewnętrznej musi rozpoczynać się linią nagłówka funkcji:

FUNCTION nazwa (parametry)

a kończyć linią końca funkcji END FUN. W treści definicji funkcji musi wystąpić co najmniej jedna instrukcja podstawienia pod nazwę funkcji. Wartość funkcji może być numeryczna lub napisowa.

Dla ilustracji — zdefiniujemy funkcję fnx jako funkcję zewnętrzną

```
1000 FUNCTION fnx (par,Y)
1010     LET wart = par * par
1020     LET pom = (wart + 1) / (wart — 1)
1030     LET fnx = pom + Y
1040 END FUN
```

Podprogramy są definiowane analogicznie jak funkcje zewnętrzne przez umieszczanie treści podprogramu między linią nagłówka:

SUBROUTINE nazwa (parametry)

a linią końca podprogramu END SUB. Dla ilustracji — przedstawimy rekurencyjny podprogram obliczania wyrazów ciągu Fibbonacciego:

```
1000 SUBROUTINE fibb (n,wart,wartn_1)
1010     IF n < = 2 THEN
1020         LET wartn, wartn_1 = 1.0
1030     ELSE
1040     CALL fibb (n_1,wartn_1,wartn_2)
1050     LET wartn=wartn_1+wartn_2
1060     END IF
1070 END SUB
```

Interesująca jest możliwość wiązania programów, które można kolejno wywoływać z aktualnie wykonywanego programu instrukcją wiązania:

CHAIN nazwa_programu (argumenty)

Nazwa programu występuje w instrukcji nagłówka programu (PROGRAM) i jest napisem.

SYSTEM PLIKÓW

Wprowadzenie programu napisanego w języku BASIC w środowisko operacyjne systemu powodowało największe lokalne rozszerzenia implementacji. Na potrzeby rozszerzonego języka BASIC uporządkowano rodzaje organizacji i typy rekordów plików, z którymi może współpracować program. Przyjęto następujący podział:

- organizacja pliku — sekwencyjna, strumieniowa, bezpośrednia, indeksowa
- typ rekordu — znakowy, wewnętrzny, rodzimy.

W minimalnym języku BASIC wykorzystywano według tej konwencji plik sekwencyjny o znakowym typie rekordu, związany na stałe z terminalem (dostępny przez instrukcje INPUT i PRINT), oraz plik strumieniowy o wewnętrznym typie rekordu, utworzony wewnątrz programu na podstawie list danych instrukcji DATA (dostępny poprzez instrukcję READ). W rozszerzonym języku stworzono możliwość deklaracji trzech rodzajów plików:

- sekwencyjnego o znakowym typie rekordu
- sekwencyjnego o wewnętrznym typie rekordu
- strumieniowego o wewnętrznym typie rekordu.

Do operacji na tych plikach wykorzystuje się instrukcje READ i WRITE. Pliki są deklarowane instrukcjami otwarcia:

OPEN # numer_logiczny: NAME nazwa, opis_pliku

Istnieją też instrukcje zamknięcia pliku CLOSE oraz wymazywania_zawartość ERASE. Powiązanie między tymi instrukcjami a danym plikiem następuje przez numer logiczny podany w instrukcji otwarcia.

INSTRUKCJE WPROWADZANIA I WYPROWADZANIA

Podane dla języka minimalnego instrukcje wprowadzania i wyprowadzania powiązano z systemem plików za pomocą odwołania do pliku przez jego numer logiczny. Instrukcję INPUT uzupełniono zwrotem PROMPT wyprowadzającym znak zachęty. Dodatkowo istnieje możliwość ustalenia czasu oczekiwania na wprowadzenie wartości (TIMEOUT liczba_sek), po przekroczeniu którego generowany jest protest (błąd). Uzupełnienie instrukcji zwrotem:

ELAPSED zmienna

umożliwia wprowadzenie do programu informacji o czasie podania wartości przez użytkownika. Wymienione rozszerzenie języka wykorzystuje się do realizacji programów sprawdzania czasu reakcji użytkowników.

Ponadto do instrukcji wydruku PRINT dołączono możliwość definiowania formatu linii wydruku (instrukcje MARGIN, ZONEWIDTH).

OPERACJE TABLICOWE

Operacje tablicowe były integralną częścią pierwotnej wersji języka. Umożliwiają one efektywne zapisanie jednorodnych działań dotyczących równocześnie wszystkich elementów tablicy. W pełnej wersji języka wprowadzono operacje na tablicach numerycznych i napisowych. Dołączono również instrukcje wprowadzania i wyprowadzania tablic za pomocą pojedynczej instrukcji, np.:

```
10 MAT INPUT tabl1, tabl$
20 MAT PRINT tabl1, tabl$
```

Dla tablic numerycznych wprowadzono operacje dodawania, odejmowania i mnożenia (oraz mnożenia przez stałą). Dodatkowo zdefiniowano standardowe funkcje algebry liniowej (TRN, INV, DET, DOT, CON, IDN, ZERO). W celu zilustrowania możliwości tych operacji podano przykłady użycia instrukcji MAT (tabl1 jest tablicą dwuwymiarową kwadratową):

— mnożenie przez tablicę o tych samych wymiarach, której wszystkie elementy są równe jedności:

```
100 MAT tabl2= tabl1 * CON
```

— mnożenie przez stałą równą 5:

```
110 MAT tabl3 = 5 * tabl2
```

— odwracanie tablicy:

```
120 MAT tabl1 = INV (tabl2)
```

Dla tablic napisowych wprowadzono ponadto możliwość konkatenacji, np.:

```
20 MAT alfa$ = alfa$ & beta$
```

oraz przypisania wartości (również pustych), np.:

```
210 MATalfa$(2:5) = („A”) & delta$
220 MAT beta$ = NULL$
```

OBSŁUGA PROTESTÓW

Protestem (ang. exception) nazywamy zgłoszenie występowania błędu podczas wykonywania programu. Pojęcie to znane było dotychczas jako przerwanie programowe, powodujące przeważnie bezwarunkowe usunięcie programu przez system operacyjny. W języku BASIC umożliwiono obsługę protestów w programie, wprowadzając mechanizm przełączania wykonywanego programu na sekwencję instrukcji, których zadaniem jest analiza i obsługa protestu.

Instrukcja warunkowa protestu (WHEN) rozpoczyna sekwencję linii podzieloną na linie instrukcji czynnych oraz linie instrukcji obsługi protestu. Sekwencje są przedzielone instrukcją użycia (USE) i zakończone instrukcją END WHEN.

W celu zilustrowania mechanizmu obsługi rozważmy poniższy przykład:

```
640 WHEN EXCEPTION IN
650     LET wart = dzielna/dzielnik
660 USE
670     PRINT „dzielnik = 0”
680     LET wart = INF
690 END WHEN
```

Linie 670, 680 zostaną wykonane tylko wtedy, gdy podczas wykonywania linii 650 wystąpi błąd (najprawdopodobniej spowodowany próbą dzielenia przez zero), a więc nastąpi zgłoszenie protestu.

Dla ułatwienia obsługi protestu istnieje funkcja podająca numer błędu (błędy są znormalizowane) oraz instrukcje powrotu do powtórnego wykonania linii, w której wystąpił błąd (RETRY), powrotu do następnej linii po linii z błędem (CONTINUE) oraz — powodująca powstanie protestu o danym numerze CAUSE.

Oprócz wyżej podanej konstrukcji istnieje możliwość zgrupowania istrukcji obsługi błędu w osobnym miejscu programu, np.:

```
1000 HANDLER dzielenie
1010     PRINT „dzielnik = 0”
1020     LET wart = INF
1030     CONTINUE
1040 END HANDLER
```

co umożliwia wykorzystywanie tych instrukcji dla każdego dzielenia, jeżeli pod każdą instrukcją z dzieleniem zapisano:

```
640 WHEN EXCEPTION IN dzielenie
650 LET wart = dzielna/dzielnik
```

* * *

Omówione w ostatniej części artykułu konstrukcje składni języka oraz ich treść semantyczna stwarzają bardzo duże możliwości strukturalizacji i modularyzacji programu. Powstaje jednak pytanie, czy jest to jeszcze język dla początkujących (Beqinners All-purpose....).

LITERATURA

[1] Draft American National Standard for BASIC. Report ANSI X3J2/82—17, American National Standards Institute, New York, October 1982

[2] Kurtz T.E.: On the Way to Standard BASIC. BYTE, pp. 182—218, June 1982.

Stały kontakt z INFORMATYKĄ gwarantuje tylko prenumerata

Zasady jej zamawiania — na czwartej stronie okładki

Blankiet zamówienia prenumeraty dla instytucji — na str. 29-30

HANNA OKTABA
Instytut Informatyki
Uniwersytet Warszawski

Prefiksowanie klasami w LOGLANIE

W poprzednim numerze INFORMATYKI została przedstawiona klasa — jedna z podstawowych konstrukcji języka LOGLAN. Jest to rodzaj modułu syntaktycznego łączącego cechy rekordu i procedury. Dla przypomnienia przytoczę schemat deklaracji klasy:

```
unit A: class(List_par_formaA);
        Dekl_lokA;
        begin
        InstrA
        end A;
```

Jest to schemat deklaracji klasy o nazwie **A**. **List_par formA** jest listą parametrów formalnych klasy **A**. **Dekl—lokA** są deklaracjami wielkości lokalnych, natomiast **InstrA** jest listą instrukcji klasy **A**. Deklaracja klasy stanowi syntaktyczny wzorzec, według którego tworzone są w programie dynamiczne jednostki zwane obiektami danej klasy. Wyrażenie generujące obiekt danej klasy przyjmuje postać:

```
new A(List_par_aktA)
```

Liczenie wartości tego wyrażenia składa się z kilku faz, związanych z rezerwacją pola pamięci dla nowego obiektu (pole dla parametrów i wielkości lokalnych), przekazaniem parametrów aktualnych oraz wykonaniem instrukcji umieszczonych w deklaracji klasy **A**. Ostateczną wartością tego wyrażenia jest adres pola pamięci utworzonego obiektu, który można przekazać jako wartość zmiennej typu **A**. Obiekty przedstawia się graficznie w postaci „pudełek", takich jak na rysunku 1.

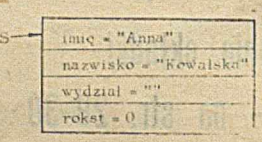
Rys. 1. Obiekt klasy A będący wartością zmiennej X typu A

Parametry oraz wielkości lokalne obiektu nazywa się atrybutami. Dostęp do atrybutów obiektu jest możliwy poprzez zmienną wskazującą dany obiekt. Jest to tzw. dostęp kropkowany, syntaktycznie i semantycznie zbliżony do dostępu pól rekordów w PASCALU. Dla zobrazowania omówionych pojęć rozpatrzymy prosty przykład:

```
unit student: class(imię, nazwisko:string);
              var wydział:string,rokst:integer;
              end student;
```

Utwórzmy obiekt klasy **student**:

```
S:=new student(„Anna", „Kowalska")
```



Rys. 2. Obiekt klasy student tuż po wygenerowaniu

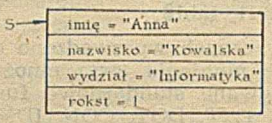
Efektem wykonania tej instrukcji będzie nowy obiekt, przedstawiony na rysunku 2, wskazywany przez zmienną **S** zadeklarowaną jako:

```
var S:student
```

Atrybuty **wydział** i **rokst** tego obiektu mają wartości równe wartościom początkowym nadawanym zmiennym w momencie deklaracji. Nadanie tym atrybutom sensownych wartości może nastąpić w wyniku wykonania następujących instrukcji:

```
S.wydział:=„Informatyka";
S.rokst:=1;
```

Po tych instrukcjach obiekt przyjmie postać pokazaną na rysunku 3.



Rys. 3. Obiekt po ustaleniu wartości wszystkich atrybutów

*

Historycznie rzecz biorąc, prefiksowanie klasami zostało po raz pierwszy wprowadzone w języku SIMULA 67. LOGLAN, adaptując pojęcie klasy, przejął wraz z nim ideę prefiksowania, uogólniając ją nieco i pozbawiając pewnych ograniczeń. Z punktu widzenia podstawowych własności prefiksowania różnice między podejściem simulowskim a loglanowskim są na tyle nieistotne, że nie będę poświęcać im tutaj miejsca.

Co to jest prefiksowanie klasami? Odpowiedź jest trudna, gdyż pojęcie to niesie za sobą wiele różnych znaczeń i zastosowań. Próbą odpowiedzi może być następujące stwierdzenie: **Prefiksowanie klasami jest metodą skrótowego definiowania modułów syntaktycznych, które są rozszerzeniami klas prefiksujących dany moduł.**
Ta trochę enigmatyczna definicja stanie się być może bardziej zrozumiała po zapoznaniu się z syntaktyczną postacią prefiksowania i jego semantycznym znaczeniem.

DEKLARACJE KLAS PREFIKSOWANYCH

Definiowanie prefiksowania klasami rozpocznę od przypadku, w którym klasa prefiksuje moduł syntaktyczny będący inną klasą. Załóżmy, że jest już zadeklarowana pewna

```
klasa o nazwie A: unit A: class(List—par—formA);
                  Dekl—lokA;
                  begin
                  InstrB
                  end B;
```

Rozszerzenie tej klasy o nowe wielkości oraz wprowadzenie nowych instrukcji może nastąpić poprzez deklarację nowej klasy, nazwijmy ją **B**, prefiksowanej klasą **A**. Sy-

ntaktyczna postać takiej deklaracji jest następująca:

```
unit B: A class(List_par_formB);
          Decl_lokB;
          begin
            InstrB
          end B;
```

Nazwa klasy prefiksującej **A** pojawia się przed słowem **class** w deklaracji klasy **B**. Deklarację klasy **B** należy rozumieć w sposób następujący: jest ona rozszerzeniem deklaracji klasy **A** o nowe parametry formalne, nowe deklaracje lokalne oraz nową listę instrukcji. Rozszerzenie listy instrukcji jest dość niekonwencjonalne. Otóż w instrukcjach klasy prefiksującej **A** może wystąpić pseudo-instrukcja **inner**, która wskazuje miejsce wykonania instrukcji klasy prefiksowanej **B**. Rozszerzenie instrukcji klasy **A** może zatem nastąpić w wybranym miejscu, wskazanym symbolem **inner**. Jeżeli pseudo-instrukcja **inner** nie występuje w treści klasy **A**, to przyjmuje się, że jest ona umieszczona jako ostatnia innstrukcja treści klasy. W tym przypadku rozszerzanie listy instrukcji klasy **A** polega na wykonaniu instrukcji **InstrB** po wszystkich instrukcjach **InstrA.**

Jako prosty przykład rozważmy deklarację następującej klasy:

```
unit rachunek: class(nr,miesiąc:integer);
               var suma:real;
               begin
               inner;
               writeln(,,Opłata za miesiąc",miesiąc);
               writeln(,,Do zapłacenia=",suma,"zł,/);
               end rachunek;
```

Jest to klasa opisująca dowolne rachunki identyfikowane numerem, stanowiące opłatę za określony miesiąc. Generowanie obiektów klasy **rachunek** nie ma głębszego sensu, gdyż we wszystkich obiektach wartości atrybutu **suma** będzie równa zero. Ukonkretnienie rachunku — tak, aby stanowił np. rachunek opłaty za elektryczność — można wyrazić następującą deklaracją klasy prefiksowanej klasą **rachunek:**

```
unit rachelektr: rachunek class(zużyciepr:integer);
                 begin
                   suma:=suma+zużyciepr*1.90;
                 end racheletr;
```

Klasa **rachelektr** stanowi rozszerzenie deklaracji klasy **rachunek** o nowy parametr **zużyciepr**; instrukcja treści tej klasy zostanie wykonana w miejscu symbolu **inner** klasy **rachunek**, dzięki czemu wartość atrybutu **suma** przyjmie pożądaną wartość.

Klasa może być prefiksem kilku różnych klas. W naszym przykładzie możemy zadeklarować odrębną klasę definiującą rachunki opłaty telefonicznej.

```
unit rachtel: rachunek class(lbrozmow:integer);
              begin
                suma:=suma+lbrozmow*2.0;
              end rachtel;
```

Klasa, która jest już prefiksowana, może być również prefiksem innej klasy. Na przykład klasę **rachelektr** możemy z kolei rozszerzyć tak, aby opisywała rachunki za elektryczność i gaz.

```
unit rachelelektrgaz: rachelelektr class(zużyciegazu:integer);
                      begin
                      suma:=suma+zużyciegazu*2.70;
                      end rachelelektrgaz;
```

Klasa **rachelelektrgaz** dodaje nowy parametr do parametrów zdefiniowanych uprzednio w klasach **rachunek** i **rachelelektr** oraz rozszerza listę instrukcji o instrukcję, która wraz z instrukcją klasy **rachelektr** zostanie wykonana w miejscu symbolu **inner** klasy **rachunek**. Dzięki temu, wartość atrybutu **suma** w momenncie drukowania będzie stanowiła sumę opłat za gaz i elektryczność.

OBIEKTY KLAS PREFIKSOWANYCH

Powróćmy do przykładu abstrakcyjnej deklaracji klasy **B** prefiksowanej klasą **A.** Na tym przykładzie wyjaśnimy sposób generowania obiektów klas prefiksowanych. Wyrażenie generujące postaci:

new B(List—par—aktA,List—par—aktB)

tworzy obiekt klasy **B**, który składa się jakby z dwu warstw przedstawionych na rysunku 4. Ponieważ obiekt klasy **B** jest rozszerzeniem obiektu klasy **A**, dlatego też lista parametrów aktualnych w wyrażeniu generującym musi zawierać parametry zdefiniowane w klasie prefiksującej **A** oraz te, które zostały dodane w deklaracji **B**. Atrybutami obiektu klasy **B** są, jak to pokazano na rysunku 4, wszystkie wielkości zdefiniowane w klasie **A** i klasie **B**.

Rys. 4. Obiekt klasy B

W trakcie obliczania wartości wyrażenia generującego:

new B(...)

wykonywane są instrukcje klasy **A** (InstrA), w których — w miejscu symbolu **inner** — wykonywane są instrukcje

klasy **B (InstrB)**. Wartością wyrażenia generującego:
new B (...)

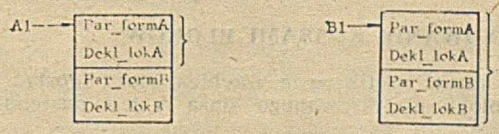
jest adres pola obiektu, który może być przekazany jako wartość zmiennej typu **B** lub typu **A**. Przyjmijmy następującą deklarację zmiennych:

var A1:A, B1:B;

Wówczas poprawne są następujące instrukcje podstawienia:

```
A1:=new B(...);
B1:=new B(...);
```

Zmienne **A1, B1** wskazują na dwa różne obiekty klasy **B**. Czy jest jakaś różnica w korzystaniu z tych zmiennych? — Tak, — różnica polega na tym, że dzięki zmiennej **B1** można dostać się przez kropkę do wszystkich atrybutów obiektu klasy **B**, natomiast zmienna **A1** pozwala na dostęp kropkowany tylko do tych atrybutów, które są w warstwie pochodzącej z klasy **A**. Rysunek 5 wyjaśnia tę sytuację graficznie.



Rys. 5. Dwa obiekty klasy B wskazywane przez zmienne A1 i B1

Nawias klamrowy obejmuje te atrybuty obiektów, które są dostępne wyrażeniami **A1.atr** lub **B1.atr.** Dostęp do pozostałych atrybutów obiektu wskazywanego przez **A1** jest możliwy, ale wymaga chwilowej zmiany typu zmiennej, co można uzyskać dzięki tzw. operatorowi zmiany kwalifikacji

qua. Napis:

A1 **qua**B. atr oznacza chwilową zmianę typu zmiennej **A1** na typ **B** i wówczas przez kropkę możemy dostać się również do atrybutów zdefiniowanych w warstwie klasy **B**. Taka zmiana typu zmiennej ma oczywiście sens tylko wówczas, gdy wartością tej zmiennej jest obiekt odpowiedniej klasy.

Przyjrzyjmy się bliżej omawianym konstrukcjom na przykładzie generowania obiektów klas definiujących różne rachunki. Przyjmijmy następujące deklaracje zmiennych:

var R1,R2,R3; rachunek;

Rozważmy instrukcję:

R1:= **new** rachelektr(125,4,250);

Zauważmy, że parametry aktualne odpowiadają parametrom formalnym **nr, miesiąc** z klasy **rachunek** i parametrowi **zużyciepr** z klasy **rachelektr.** W trakcie generowania tego obiektu zostaną wykonane instrukcje w ten sposób, że najpierw będzie policzona wartość atrybutu **suma** (zamiast symbolu **inner** w klasie **rachunek**), a następnie zostaną wydrukowane odpowiednie informacje. Wykonanie instrukcji:

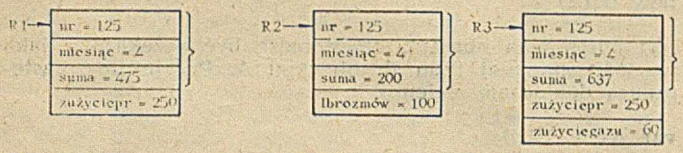
R2:=**new** rachtel(125,4,100);

spowoduje powstanie obiektu klasy **rachtel,** którego instrukcje będą wykonane analogicznie jak w obiekcie klasy **rachelektr.**

Wreszcie rozważmy instrukcję:

R3 :=**new** rachelektrgaz(125,4,250,60);

Lista parametrów aktualnych uwzględnia kolejno parametry z klas **rachunek, rachelektr** i **rachelektrgaz.** Lista instrukcji tego obiektu będzie zrealizowana w następujący sposób: w miejsce symbolu **inner** klasy **rachunek** wejdą najpierw instrukcje klasy **rachelektr,** potem klasy **rachelektrgaz,** a następnie zostaną wykonane końcowe instrukcje z klasy **rachunek.**



Rys. 6. Obiekty klas rachelektr. rachtel, rachelektrgaz

Rysunek 6 przedstawia wszystkie trzy obiekty wygenerowane powyższymi instrukcjami. Klamrami są zaznaczone te atrybuty obiektów, które można odczytać przez kropkę poprzez zmienne **R1, R2, R3,** bez zmiany typu tych zmiennych. Co oznacza, że poprawne są wyrażenia: **R1.nr, R2.miesiąc, R3.suma.** Pozostałe atrybuty można odsłonić zmieniając typ zmiennych, np.:

R1**qua**rachelektr,zużyciepr,
R2**qua**rachtel.lbrozmów,
R3**qua**rachelektrgaz.zużyciegazu.

PREFIKSOWANIE KLASAMI BLOKÓW

Klasy mogą prefiksować również bloki. Postać syntaktyczna bloku prefiksowanego klasą **A** jest następująca:

```
pref A(List_par_aktA) block
                        Decl_lok;
                        begin
                          Instr
                        end
```

Znaczenie prefiksowania bloków jest trochę odmienne od prefiksowania klas, co wynika z faktu, że deklaracja bloku jest jednocześnie instrukcją. W prefiksowaniu bloku pojawia się nazwa klasy wraz z listą parametrów aktualnych, co ma oznaczać utworzenie konkretnego obiektu klasy **A,** w kontekście którego będą wykonywane instrukcje bloku. Instrukcje te zostaną wykonane w miejscu symbolu **inner** występującego w treści klasy **A.** Blok prefiksowany może w swoich instrukcjach korzystać w sposób bezpośredni ze wszystkich atrybutów zdefiniowanych w klasie prefiksującej. Bloki prefiksowane są najczęściej używane w następującej formie:

- klasa prefiksująca stanowi definicję złożonej struktury danych, np. drzew binarnych czy kolejek priorytetowych, lub stanowi definicję pewnego systemu pomocnego przy rozwiązywaniu konkretnego rodzaju zadań, np. symulacyjnych
- blok jest rozwiązaniem konkretnego zadania, które posiłkuje się narzędziami zdefiniowanymi w prefiksie jako wygodną strukturą pomocniczą.

*

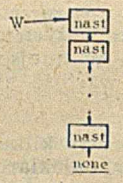
Idee prefiksowania klasami można najłatwiej zrozumieć analizując przykłady. Proponuję prześledzenie jednego z nich. Zaczniemy od deklaracji klasy definiującej strukturę danych stosu. Następnie zastosujemy tę strukturę jako pomocniczą w programie, w którym będziemy mieli do czynienia zarówno z prefiksowaniem klasą innej klasy, jak i z prefiksowaniem klasą bloku.

Deklaracja klasy STOS

Strukturę danych stosu można w LOGLANIE definiować na wiele różnych sposobów. Zaznaczam, że przedstawiona wersja będzie dość prymitywną, niemniej wystarczająca do takich celów, powiedzmy — dydaktycznych.

```
unit STOS: class;
     unit selem: class;
                 var nast:selem;
                 end selem;
     var W: selem; (* W-wierzchołek stosu * )
     unit połóż: procedure(X: selem) ; ( *połóż X na wie-
     rzchołku stosu * )
                 begin
                   if W=none then W:=X;
                             else X.nast.=W;
                                  W:=X;
                   fi
                 end połóż;
     unit usuń: procedure; ( * usuń element z wierzchołka
     stosu *)
                 begin
                   if W≠none then W:=W.nast fi
                 end usuń;
end STOS;
```

Klasa **STOS** definiuje stos, którego elementami będą obiekty klasy **selem.** Obiekty te będą połączone w ciąg dowiązaniami **nast.** Wyróżniony element tego ciągu, zwany wierzchołkiem stosu, będzie wskazywany zmienną **W.** Rysujak 7 pokazuje przykładowy ciąg obiektów reprezentujących stos.



Rys. 7. Ciąg obiektów klasy selem reprezentujących stos

Klasa **STOS** definiuje również dwie operacje na stosie nazwane **połóż** i **usuń,** których znaczenie jest standardowe.

Prefiksowanie klasą STOS

Zastosujemy klasę **STOS** w programie rozwiązującym znane zadanie zamiany wyrażeń arytmetycznych zapisanych w notacji infiksowej (operator pomiędzy argumentami) na równoważne wyrażenia w notacji postfiksowej, tzw. odwrotnej notacji polskiej (np. **(a+b)*c** na **ab+c***). Dla uproszczenia akcji związanych z rozpoznawaniem danych, przyjmiemy parę założeń dotyczących postaci wyrażenia wejściowego. Załóżmy więc, że w wyrażeniu tym mogą występować tylko zmienne jednoliterowe, nie występują stałe liczbowe, występują tylko operatory dwuargumentowe: +, —, * ,/ , ↑ (potęgowanie), wyrażenie może mieć postać nawiasową (przy czym przyjmiemy, że całe wyrażenie musi być ujęte w nawiasy). Ciąg danych kończy znak $.

Algorytm rozwiązujący zadanie posługuje się stosem jako strukturą pomocniczą do zapamiętywania operatorów. Prawdopodobnie algorytm ten jest wszystkim dobrze znany, ale przypomnę go w skrócie dla wyrażeń spełniających przyjęte założenia. W algorytmie istotną rolę odgrywają priorytety operatorów.

Przyjmiemy następujące priorytety: dla + — priorytet 1, dla *, / — 2, dla ↑, —3, ponadto znak ' (' będzie miał priorytet 0.

Algorytm

Wczytaj znak c. W zależności od postaci c, wykonaj następujące akcje:

- c=' $' — koniec programu
- c jest literą — wydrukuj tę literę
- c=' (' — połóż informacje o ,(' na stos
- c=')' — usuń ze stosu informacje o operatorach, aż do informacji o pierwszym napotkanym nawiasie ,(', usuwane operatory o priorytetach większych lub równych priorytetowi c (z równoczesnym ich drukowaniem), na koniec połóż informację o operatorze c na stos.

Przedstawiam teraz program rozwiązujący postawione zadanie.

Program

```
nit STOS: class; (* deklaracja klasy STOS*) ... end STOS;
pref STOS block (* blok prefiksowany, w którym dostępne są
                   wszystkie atrybuty klasy STOS *)
  unit symbol: selem class(znak:char,prior:integer);
                 end symbol;
  var c:char;
  begin (*początek instrukcji bloku*)
    do
      read(c); ( * wczytanie kolejnego znaku danych *)
      if c = ' $' then exit (*wyjście z pętli—koniec danych*) if;
      if letter(c) then (*funkcja boolowska letter bada, czy znak
                          c jest literą* )
                        write(c)
                 else
      if c=' (' then call połóż(new symbol(c,0) )
                 else
      if c=') ' then while Wquasymbol prior≠0
                 do
                   write(Wquasymbol.znak);
                   call usuń;
                 od;
                   call usuń: (*usunięcie ,( ' bez wypisania *)
                   else (*c jest operatorem*)
                     block
                     var pr:integer; (*pomocnicza zmienna do
                                       obliczania priorytetu c*/
                     begin
                       if c=+, orif c= '—' then pr: =1 else
                         if c= '*' orif c= '/' then pr: =2 else
                                                    pr : =3
                       fi
                       fi;
                       while Wquasymbol,prior≥ pr
                       do
                         write(Wquasymbol.znak) ;
                         call usuń;
                       od;
                       call połóż(new symbol(c,pr) ) ;
                       end (* podbloku *)
                 fi fi fi ;
               od
    end (* bloku prefiksowanego *)
end (* programu *)
```

W programie tym dwukrotnie zastosowano mechanizm prefiksowania klasami. Po raz pierwszy — gdy klasa **STOS** prefiksuje blok, a po raz drugi — gdy klasa **selem** prefiksuje klasę **symbol**. Już na tym przykładzie można się zorientować, że ten sam mechanizm może służyć różnym celom. Klasa **STOS**, prefiksująca blok, dostarcza pomocniczej struktury danych dla algorytmu zapisanego w instrukcjach bloku. Natomiast klasa **symbol**, zdefiniowana w bloku i prefiksowana klasą **selem**, stanowi rozszerzenie bazowej struktury elementu stosu o informacje potrzebne w konkretnym algorytmie. Łatwo można sobie teraz wyobrazić inny program, w którym używany jest ten sam stos (w sensie deklaracji klasy **STOS**), ale elementy trzymane na stosie niosą ze sobą zupełnie inne informacje zdefiniowane w innej podklasie klasy **selem**.

Klasy w LOGLANIE mogą być prefiksami również funkcji i procedur. Wspominam o tym tylko gwoli informacji, gdyż dokładniejsze omówienie tebo tematu wymagałoby przedstawienia sensownego przykładu, co mogłoby znużyć Czytelników.

* * *

Prefiksowanie klasami jest silnym mechanizmem umożliwiającym przede wszystkim czytelną modularyzację programów. Metoda zstępująca (top-down) budowania programów może być przy użyciu prefiksowania zrealizowana w sposób prosty i naturalny. Dzięki klasom i prefiksowaniu możliwe jest abstrakcyjne definiowanie struktur danych, czy nawet języków problemowo-zorientowanych, a następnie ukonkretnienie ich dla poszczególnych zastosowań. Prefiksowanie jest szczególnie atrakcyjne w połączeniu z oddzielną kompilacją klas. Wówczas korzystanie z biblioteki klas definiujących całe systemy polega na prostym prefiksowaniu klasami odpowiednich bloków czy klas w naszym programie. Jednym słowem — jest to bardzo interesujący mechanizm, nadal niedoceniony i chyba niedostatecznie znany i rozumiany.

KSIĄŻKI NADESŁANE

WYDAWNICTWA NAUKOWO-TECHNICZNE

Bolc L., Cichy M., Różańska L.: Przetwarzanie języka naturalnego. Seria BIO, nakład 3000 egz., 167 str., cena 110 zł, Warszawa 1982.

Książka zawiera opis wybranych metod automatycznego przetwarzania tekstów języka naturalnego. Autorzy przedstawili konkretne rozwiązania umożliwiające stosowanie tych metod w praktyce. W pracy znaleźć można dość szczegółową charakterystykę systemu statystycznego przetwarzania tekstów COCOA oraz dwóch systemów przetwarzania informacji LUNAR i REL z dostępem w języku naturalnym. Omówiona została ponadto problematyka reprezentacji wiedzy w systemach z dostępem w języku naturalnym, w tym m.in. języki reprezentacji wiedzy KRL, FRL i OWL.

Alagić S., Arbib M.: Projektowanie programów poprawnych i dobrze zbudowanych. Seria BIO, nakład 5000 egz., 294 str., cena 175 zł, Warszawa 1982.

Książka prezentuje techniki zastępującego projektowania programów oraz ich formalnej weryfikacji. Oprócz klasycznych reguł Hoare'a dowodzenia poprawności programów zawierających podstawienia, instrukcje warunkowe i pętle — opisano w niej reguły dla skoków oraz procedur i funkcji (także rekurencyjnych). Materiał książki zawiera wiele przykładów algorytmów, wzbogacony jest również o liczne zadania. Ponadto książka może służyć jako podręcznik do nauki języka PASCAL.

Fortuna Z., Macukow B., Wąsowski J.: Metody numeryczne. Seria EIT, nakład 10 000 egz., 369 str., cena 120 zł, Warszawa 1982.

Książka stanowi podręcznik akademicki omawiający zagadnienia interpolacji, aproksymacji, rozwiązywania równań nieliniowych i ich układów, całkowania numerycznego, rozwiązywania układów algebraicznych równań liniowych, obliczania wartości i wektorów własnych macierzy, a także pewne zagadnienia związane z równaniami różniczkowymi. Algorytmy zapisane zostały bądź za pomocą schematów blokowych, bądź w FORTRANIE. Zamieszczono także ocenę efektywności wielu algorytmów.

Poniższy artykuł nawiązuje do opublikowanej w INFORMATYCE nr 1/83 propozycji sprzęgu wewnątrzkasetowego KRABUS. Sprawa wyboru sprzęgu dla systemu mikrokomputerowego ma podstawowe znaczenie z technicznego punktu widzenia, ale nie tylko, gdyż standaryzacja magistrali, złącz i płytek może przynieść w skali kraju ogromne oszczędności. Dlatego chętnie podtrzymamy dyskusję na ten temat, stwarzając forum wymiany poglądów.

Problem jest jednak niezwykle złożony i trudno będzie wybrać optymalne rozwiązanie, nawet w teorii, a cóż dopiero w praktyce. Wystarczy powiedzieć, że w Stanach Zjednoczonych trwają równoległe prace normalizacyjne nad czterema typami magistrali (sprzęgów) o zasadniczo różnych koncepcjach (wspominają o nich także autorzy artykułu). Wskutek rozdźwięków w łonie jednej z grup normalizacyjnych, najbardziej pożądany projekt tzw. FUTURE BUS (P-896) rozpadł się na dwa inne — amerykański i europejski (I-896).

Z kolei w Centrum Badań Jądrowych CERN pod Genewą zalecono stosowanie jeszcze innych norm — G-64 i VME (odpowiednio, dla mikrokomputerów 8- i 16-bitowych). Warto o tym wiedzieć, choćby dlatego, że pięcioletnie zapotrzebowanie na systemy mikrokomputerowe tylko w tej jednej organizacji ocenia się na 37 tys. modułów i 2 tys. kaset. Podobne względy zadecydowały o tym, że CERN miał poważny udział w rozwoju standardu znanego jako CAMAC (p. INFORMATYKA, nr 4—5/1982).

Wydaje się, że duże znaczenie ma rozpoczęcie publicznej dyskusji na ten temat w kraju. Spodziewamy się dalszych artykułów i będziemy publikować wszelkie rozsądne uwagi.

Redakcja

MARIUSZ POSTÓŁ, JERZY STANIASZCZYK
WOJCIECH WEBER, ANDRZEJ ZDANOWSKI
Łódź

Sprzęgi systemowe w urządzeniach mikrokomputerowych

Krajowi producenci urządzeń mikrokomputerowych opracowali własne sprzęgi systemowe (wewnątrzkasetowe) służące do łączenia poszczególnych modułów mikroprocesorowych. Różnią się one wzajemnie tak bardzo, że moduły wytwarzane w kraju przez różnych producentów nie mogą bezpośrednio współpracować. Moduły mogą być wykorzystywane tylko w tych urządzeniach, do których były projektowane.

Brak gotowych modułów wyposażonych w powszechnie przyjęty sprzęg zmusza producentów urządzeń do pracy od podstaw. W poszczególnych ośrodkach każdorazowo przystosowuje się znane już rozwiązania do własnego standardu sprzęgu lub do posiadanych aktualnie typów kaset i złącz.

Powszechny staje się pogląd, że taka sytuacja jest wynikiem braku krajowej normy sprzęgu systemowego dla urządzeń mikrokomputerowych. W wielu ośrodkach rozpoczęto prace nad stworzeniem uniwersalnego, a przez to najlepszego i „jedynego" rozwiązania, które stając się krajową normą zaspokoiłoby oczekiwania wszystkich. Wynikiem tych prac są zwykle sprzęgi będące zlepkiem pomysłów zaczerpniętych z rozpowszechnionych na świecie standardów. Przykładem takiego rozwiązania jest KRABUS [5].

ZAGADNIENIE WYBORU SPRZĘGU

Dążenie do wprowadzenia uniwersalnego sprzęgu przysłania rzeczywiste problemy związane ze standaryzacją modułów mikroprocesorowych. Powodem narastających problemów nie jest bowiem brak jednego sprzęgu. Uniwersalny sprzęg nadający się do wszystkich zastosowań niestety nie istnieje. Problem polega na rozpowszechnieniu w kraju niewłaściwych sprzęgów. Za takie należy uznać wszystkie sprzęgi, które różnią się od przyjętych na świecie standardów lub są ich modyfikacją. Przyjęcie jednego z nich jako normy krajowej rozwiązałoby wprawdzie wiele z przedstawionych na wstępie problemów, ale spowodowałoby powstanie w przyszłości identycznej sytuacji w wymianie handlowej z innymi krajami. Prowadziłoby to do oczywistej izolacji naszego kraju w dziedzinie mikroelektroniki.

Przy wyborze sprzęgów, które powinny być zalecane do określonych zastosowań, należy brać pod uwagę:

- ich popularność na świecie
- możliwość łatwej współpracy z krajowymi układami scalonymi
- dostępność stosowanych kaset i złącz.

Sprzęgi spełniające powyższe wymagania należy rozpowszechniać bez żadnych zmian, jak np. sprzęg IEC AMS.

Istnieje jednak duża grupa sprzęgów, które spełniają tylko dwa pierwsze wymagania, jak np.: [1] IEEE 796 (MULTIBUS), IEEE 696 (S—100) i STD-Bus (IEEE P961). Rozpowszechnienie ich w kraju będzie możliwe dopiero po podjęciu decyzji o produkcji odpowiednich kaset i złącz lub po wprowadzeniu zmian w standardzie mechanicznym. Pierwsze rozwiązanie jest lepsze, lecz wymaga

środków i czasu na osiągnięcie zamierzonego celu. Korzyścią tego wariantu jest możliwość eksportu urządzeń mikrokomputerowych. Przyjmując to rozwiązanie trzeba mieć pewność, że krajowi producenci będą w stanie wytwarzać sprzęt mikrokomputerowy na światowym poziomie.

Drugie rozwiązanie polegające na wprowadzeniu zmian w standardzie mechanicznym jest tańsze. Moduły zachowujące tylko standard logiczny i elektryczny określonego sprzęgu mogą współpracować bezpośrednio z modułami spełniającymi wszystkie jego wymagania (także — mechaniczne). Można to osiągnąć przez dodatkowe połączenie styków złącza stosowanego w rozwiązaniu zmodyfikowanym z odpowiednimi stykami złącza zalecanego w normie. Takie rozwiązanie przekreśla szanse na eksport urządzeń mikrokomputerowych, lecz umożliwia wykorzystywanie doświadczenia firm zagranicznych.

PRZYKŁADOWE ROZWIĄZANIA

W tabeli przedstawiono przykładowy sprzęg systemowy, który zachowuje standard elektryczny i logiczny sprzęgu MULTIBUS [2, 4, 6]. W proponowanym rozwiązaniu sygnały sprzęgu MULTIBUS odpowiadają stykom dwóch złącz pośrednich typu 811/821 (produkcji zakładów ELTRA).

Przykładowe rozwiązania sprzęgów opartych na standardach MULTIBUS i IEEE 796

Sygnały sprzęgu MULTIBUS		Styk	Sygnały sprzęgu IEEE 796	
b	a		b	a
+ 5 V	+ 5 V	1	+ 5 V	+ 5 V
GND	GND	2	GND	GND
+ 12 V	− 12 V	3	+ 12 V	− 12 V
BCLK	CCLK	4	BCLK	CCLK
BPRN	BPRO	5	BPRN	BPRO
BUSY	BREQ	6	BUSY	BREQ
MRDC	MWTC	7	MRDC	MWTC
IORC	IOWC	8	IORC	IOWC
XACK	—	9	XACK	—
INH 1	INH 2	10	INH 1	INH 2
BHEN	CBRQ	11	BHEN	CBRQ
INIT	INTA	12	INIT	INTA
INT 0	INT 1	13	INT 0	INT 1
INT 2	INT 3	14	INT 2	INT 3
INT 4	INT 5	15	INT 4	INT 5
INT 6	INT 7	16	INT 6	INT 7
ADR 0	ADR 1	17	A 0	A 1
ADR 2	ADR 3	18	A 2	A 3
ADR 4	ADR 5	19	A 4	A 5
ADR 6	ADR 7	20	A 6	A 7
ADR 8	ADR 9	21	A 8	A 9
ADR A	ADR B	22	A 10	A 11
ADR C	ADR D	23	A 12	A 13
ADR E	ADR F	24	A 14	A 15
AD 10	AD 11	25	A 16	A 17
AD 12	AD 13	26	A 18	A 19
—	—	27	—	—
—	—	28	—	—
DAT 0	DAT 1	29	D 0	D 1
DAT 2	DAT 3	30	D 2	D 3
DAT 4	DAT 5	31	D 4	D 5
DAT 6	DAT 7	32	D 6	D 7
DAT 8	DAT 9	1	D 8	D 9
DAT A	DAT B	2	D 10	D 11
DAT C	DAT D	3	D 12	D 13
DAT E	DAT F	4	D 14	D 15
—	—	...		
− 5 VB	− 5 V	22		
+ 15 V	− 15 V	23		
WALT	HALT	24		
PAR 1	PAR 2	25		
ALE	RESET	26		
MPRO		27		
PFSN	PFSR	28		
ACLO	PFIN	29		
+ 12 VB	− 12 VB	30		
GND		31		
+ 5 VB	+ 5 VB	32		

Złącza te są umieszczone w kasecie przystosowanej do wkładania modułów na tzw. Eurokarcie o szerokości 233,4 mm.

Złącze górne zawiera wszystkie sygnały szyny adresowej i szyny sterującej oraz 8 sygnałów szyny danych. Ten zbiór sygnałów jest wystarczający dla mikroprocesorów 8-bitowych. Złącze górne zawiera także 3 napięcia zasilania i 5 styków rezerwowych.

Na złączu dolnym umieszczono pozostałe sygnały szyny danych i sygnały ze złącza P2 sprzęgu MULTIBUS. Złącze dolne zawiera także napięcie zasilania —5 V oraz 35 styków rezerwowych.

W tej samej tabeli przedstawiono przykładowe rozwiązanie zachowujące standard elektryczny i logiczny sprzęgu IEEE 796 [3]. Jest on najnowszą wersją sprzęgu MULTIBUS zalecaną przez IEEE (Institute of Electrical and Electronics Engineers, USA).

W normie IEEE 796 zmieniono nazwy sygnałów, zrezygnowano z napięcia zasilania —5 V i nie zdefiniowano sygnałów ze złącza P2 sprzęgu MULTIBUS. W przyszłości przewiduje się wprowadzenie uzupełnień do tej normy. Cztery rezerwowe styki na złączu górnym (b27, a27, b28, a28) są przeznaczone na dodatkowe sygnały szyny adresowej, a piąty styk (a9) służy do rozszerzenia zbioru sygnałów sterujących.

Moduły zgodne ze sprzęgiem MULTIBUS mogą bezpośrednio współpracować z modułami wyposażonymi w sprzęg IEEE 796.

LITERATURA

[1] Bailey Ch.: Bus Standards Enrich Microcomputer Board Variety. Electronic Design, April 1982

[2] Barthmaier J.: INTEL Multibus Interfacing. Aplication Note, Intel Corporation, Santa Clara, CA, 1979

[3] Boberg R. W.: Proposed Microcomputer System 796 Bus Standard. Computer, pp. 89—105, October 1980

[4] Harr T. A., Phillips R.: Interrupts Call the Shots in Scheme Using Two Microprocessors. Electronics, pp. 166—170, February 1980

[5] Jankowski M. T.: KRABUS — sprzęg wewnątrzkasetowy dla systemów mikrokomputerowych. INFORMATYKA, nr 1, 1983, str. 17

[6] Ogdin C. A.: μC Bus Standardization Makes (Dollars and) Sense. EDN, pp. 77—83, April 1978.

Zjednoczone Zakłady
Przemysłu Odzieżowego

„WÓLCZANKA"

oferują do sprzedaży
maszyny w bardzo dobrym stanie:

Dziurkarkę Soemtron (1 szt.)

Sprawdzarkę Soemtron (1 szt.)

Oferty prosimy kierować pod adresem:

ZZPO „WÓLCZANKA" ul. Wólczańska 243,
93-035 Łódź, tel. 447-84

EO/758/K/83

Pamięci dyskowe typu Winchester

Technikę zapisu i odczytu, stosowaną w pamięciach dyskowych, a znaną jako technika Winchester, opracowano w firmie IBM w r. 1968. Specyficzną cechą tej techniki jest sposób ustawiania głowicy na właściwej (w tym przypadku rzędu ułamka mikrometra) wysokości lotu. Zespół głowic i dysków stanowi oddzielną konstrukcyjnie jednostkę. W stanie spoczynkowym głowice, przymocowane do ramienia pozycjonera [1]), leżą na powierzchniach dysków. Po włączeniu napędu i wprawieniu dysków w ruch obrotowy, głowice ślizgają się po powierzchniach dysków (powierzchnie te są pokryte specjalną substancją o bardzo małym współczynniku tarcia). Przy odpowiedniej prędkości obrotowej głowice, pod wpływem działających przypowierzchniowych sił aerodynamicznych, unoszą się na żądaną wysokość (0,5—1,0 μm).

Pamięci dyskowe, w których stosuje się technikę Winchester do ustawiania wysokości lotu głowic, zwane pamięciami dyskowymi typu Winchester, pojawiły się na rynku komputerowym w pierwszej połowie lat siedemdziesiątych.

Początkowo produkowano pamięci z wymiennym pakietem dysków, ale okazało się to nieopłacalne. Koszt pakietu, zawierającego zespół dysków i głowic, oraz koszt mechanizmu napędowego (bardzo wysokie wymagania dotyczące tolerancji wykonania elementów, warunkujących centryczne i powtarzalne osadzenie pakietu dysków na wspólnej osi) były w sumie zbyt duże i nie były rekompensowane przez zwiększenie pojemności i polepszenie niezawodności pamięci.

Po pewnym czasie rozpoczęto produkcję pamięci typu Winchester z niewymiennymi pakietami głowic i dysków. Okazało się, że takie rozwiązanie, ze względu na dużo niższą cenę, cieszy się większym powodzeniem. W 1979 pamięć Winchester z niewymiennym pakietem dysków, o pojemności do 60 M bajtów i wadze ok. 10 kg, kosztowała ok. 1500 $. Kolejno wprowadzano pamięci o coraz mniejszych średnicach dysków. Początkowo były to dyski o średnicy 14 cali (355,6 mm), później 8 cali (203,2 mm), a ostatnio produkuje się również pamięci o średnicy dysku 5 1/4 cala (133,35 mm).

Podstawowe parametry pamięci dyskowych

Parametr	IBM 3300	Dysk elastyczny 8 cali	Dysk Winchester 8 cali	Dysk Winchester 14 cali
Pojemność [M bajt]	25—300	0,4—1,6	5—15	15—60
Średni czas dostępu [ms]	35—40	100—200	60—70	50—70
Gęstość zapisu [bit/cm]	1575—2362	1260—2520	1968—2756	2165—2953
Gęstość ścieżek [liczba ścieżek/cm]	78,7—157	19—38	67—118	67—118
Średnia szybkość przesyłania [M bit/s]	7—9	0,25—0,5	4—6	7—9
Średni czas między uszkodzeniami [1000 h]	4—6	5—8	8—12	8—12

[1]) Pozycjoner (przesuwnik głowic) jest ruchomym podzespołem umożliwiającym ustawienie przymocowanych do niego głowic nad żądanymi ścieżkami

Pamięci typu Winchester mają bardzo dobre parametry użytkowe, a także dużą niezawodność oraz dość niską cenę w porównaniu z innymi pamięciami dyskowymi podobnej klasy. Prosta budowa głowic wpływa korzystnie na niezawodność działania oraz zmniejszenie ciężaru. Ze względu na bardzo małą odległość między głowicą i dyskiem uzyskuje się znaczną gęstość zapisu. Szczelne zamknięcie głowicy wraz z dyskami we wspólnej komorze minimalizuje dostęp wszelkiego typu zanieczyszczeń, co jest niezmiernie istotne. Dlatego też zespół głowic i dysków zamyka się w specjalnej komorze, wyposażonej w bardzo wydajne filtry powietrzne. Podstawowe parametry pamięci Winchester oraz innych typów pamięci dyskowych zestawiono w tabeli.

KONSTRUKCJA MECHANICZNA

W nowych rodzajach pamięci (opracowanych przez firmę RODINE LTD), pracujących z dyskami o średnicy 5 1/4 cala, zastosowano wiele interesujących rozwiązań zarówno mechanicznych, jak i elektrycznych. Dzięki temu pamięci te mają bardzo duże pojemności (w porównaniu z innymi pamięciami o takiej samej średnicy dysków), krótki czas dostępu, a także bardzo dużą niezawodność i szerokie możliwości autodiagnostyczne.

W okresie, kiedy w pamięciach Winchester o średnicy dysku 5 1/4 cala stosowano powszechnie dwa talerze dyskowe, firma RODINE LTD wprowadziła zespół składający się z czterech talerzy co — przez zwiększenie powierzchni nośnika — znacznie powiększyło pojemność pamięci. Talerze są zamontowane na wspólnej, specjalnej osi, a ich wzajemne odległości wynoszą 1/4 cala (63,5 mm); w niektórych rodzajach pamięci — 0,3 cala (76,2 mm).

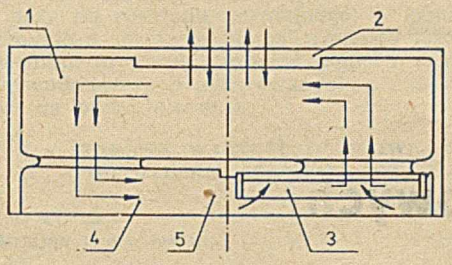
Zmniejszenie odległości między talerzami spowodowało konieczność nowego rozwiązania rozpływu powietrza w szczelnej, wspólnej komorze. Chodzi o to, aby wszędzie nad powierzchniami dysków przepływ powietrza był laminarny, a nie turbulentny. Dzięki zastosowaniu we wspólnej osi specjalnych szczelin odpowietrzających, przepływ powietrza jest laminarny w obszarach, w których w przeciwnym przypadku, byłby turbulentny oraz w których występowałoby podciśnienie. Głowice są przy tym umieszczone na ramionach pozycjonera w taki sposób, że leżą poza turbulentnym obszarem powietrza wywołanym przez te ramiona. Jest to szczególnie korzystne w czasie używania ścieżek wewnętrznych, gdyż w tym obszarze odstęp głowic od talerzy jest najmniejszy. Laminarny przepływ powietrza zapewnia utrzymywanie właściwego odstępu głowic od talerzy oraz eliminuje ewentualne drżenie głowic.

Niesłychanie istotnym czynnikiem, wpływającym na jakość pracy, jest niezawodność oraz dokładność pracy mechanizmu pozycjonera. Można umieszczać go wewnątrz lub na zewnątrz szczelnej komory zawierającej dyski i głowice. Za umieszczeniem go na zewnątrz przemawia fakt, iż jest on źródłem zanieczyszczeń mechanicznych. Jednocześnie zaś zapewnienie właściwego stopnia czystości jest konieczne dla niezawodnej pracy tego podzespołu. Spotyka się więc zarówno pamięci dyskowe, w których zastosowano pierwszą metodę, jak i takie, w których wykorzystano drugą.

Pamięci firmy RODINE LTD charakteryzuje nowe, kompromisowe i jak dotąd unikalne rozwiązanie. Zastosowano w nich bowiem układ dwukomorowy. Pozycjoner znajduje

się we własnej, szczelnej komorze (komora pozycjonera), która jest oddzielona od równie szczelnej komory mieszczącej dyski i głowice (komora głowic i dysków). Obydwie komory są wzajemnie połączone za pośrednictwem odpowiednich filtrów. Powietrze jest przepompowywane z komory głowic i dysków do komory pozycjonera, a następnie z powrotem poprzez specjalny, bardzo wydajny filtr. Zastosowanie filtru stworzyło warunki przepływu powietrza między komorami przy jednoczesnej separacji głowic i dysków od zanieczyszczeń wytwarzanych w komorze pozycjonera. Powietrze opływa komory z szybkością 5 razy/min, a filtr eliminuje zanieczyszczenia wielkości ok. 0,5 μm ze sprawnością 99,97%.

Układ dwukomorowy rozwiązuje dodatkowo problem kompensacji termicznej. Powietrze, opływające głowice i dyski oraz pozycjoner, wyrównuje temperatury pracy tych podzespołów. Na rysunku 1 przedstawiono schematycznie układ dwukomorowy oraz zaznaczono kierunki przepływu powietrza. Zmiany temperatury pracy, wskutek występowania zjawiska rozszerzalności termicznej, mogą wywołać niewłaściwe położenie głowic w stosunku do ścieżek. Można to wyeliminować stosując odpowiednie materiały oraz kształty geometryczne elementów składowych pozycjonera. W omawianej pamięci dyskowej uzyskano kompensację termiczną położenia głowic w stosunku do ścieżek z dokładnością 1 μcal/°C w całym, 40-stopniowym przedziale temperatur pracy.

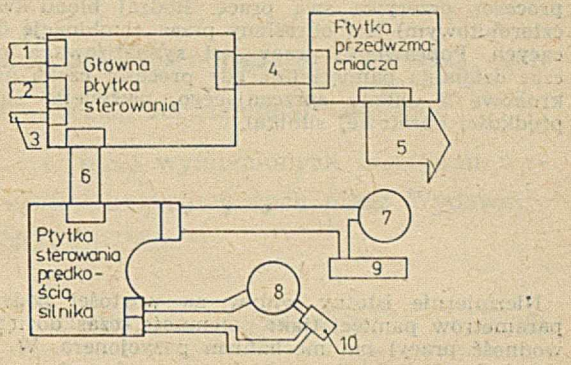


Rys. 1. Schemat układu dwukomorowego, oddzielającego pozycjoner od dysków i głowic (1 — komora dysku, 2 — filtr wyrównujący ciśnienie, 3 — filtr w układzie cyrkulacji powietrza, 4 — strumień powietrza, 5 — komora pozycjonera)

Uszczelniony układ dwukomorowy musi zawierać specjalny filtr, utrzymujący w komorach ciśnienie równe ciśnieniu atmosferycznemu. Niezmiernie istotną sprawą jest przy tym właściwa (centralna) lokalizacja takiego filtru, bagatelizowana niekiedy przez producentów. Wskutek ruchu obrotowego dysków zwiększa się bowiem masa powietrza przy ich zewnętrznych krawędziach, a zmniejsza — przy osi. Jeśli filtr odpowietrzający znajdowałby się daleko od osi komory głowic i dysków, odprowadzałby powietrze na zewnątrz, co z kolei powodowałoby spadek ciśnienia przy osi dysków do wartości niższej niż zewnętrzne ciśnienie atmosferyczne. Jednocześnie zaś łożyska napędowe, leżące wzdłuż osi komory głowic i dysków, są w kontakcie z powietrzem zewnętrznym o ciśnieniu atmosferycznym. Powstała w ten sposób różnica ciśnień wywoływałaby wsysanie smaru z łożysk napędowych do komory głowic i dysków. Filtr musi w związku z tym znajdować się wokół osi komory głowic i dysków, eliminując w ten sposób możliwość powstania niepożądanej różnicy ciśnień. Kanały powietrzne prowadzone w podzespole osi dysków umożliwiają wyrównywanie się ciśnienia wewnątrz komory.

UKŁADY STEROWANIA

Układy elektroniczne pamięci Winchester znajdują się na trzech płytkach: przedwzmacniacza odczytu, sterowania prędkości silnika oraz głównego układu sterowania (rys. 2). Ponieważ przedwzmacniacz odczytu i diody izolacyjne głowicy powinny przy tym znajdować się możliwie jak najbliżej głowic, ale nie w podzespole głowicy, montuje się je obok podzespołu głowicy i łączy z nim giętką wiązką przewodów. W kanale odczytu, pierwszy stopień wzmacniacza sygnałów, dochodzących z głowicy, znajduje się na płytce przedwzmacniacza odczytu. Dzięki temu uzyskuje się niski poziom szumu przy dużym wzmocnieniu oraz możliwości wyboru, a także separacji głowic.



Rys. 2. Schemat układu sterowania pamięci Winchester (1 — szyna danych, 2 — szyna sterowania, 3 — przewody zasilania, 4 — wtykowy zespół połączeniowy, 5 — do głowic zapisu, 6 — płaski kabel połączeniowy, 7 — silnik krokowy, 8 — silnik napędu głównego, 9 — detektor ścieżki zerowej, 10 — hamulec indukcyjny

Płytka sterowania prędkością silnika zawiera fazoczuły układ sprzężenia zwrotnego, który stabilizuje prędkość obrotów silnika głównego napędu. Znajdują się na niej ponadto układy tłumienia częstotliwości radiowych oraz układy ograniczające prąd silnika krokowego, jak również układy pełniące funkcje zasilacza dla hamulca elektromagnetycznego i detektora ścieżki zerowej. Silnik głównego napędu jest dwufazowym silnikiem bezszczotkowym. Pracuje z prędkością 3600 obr/min i wytwarza w trakcie każdego obrotu dwa impulsy, doprowadzane następnie do układu sprzężenia zwrotnego. Impulsy te są generowane przez zawarty w silniku układ scalony generatora Halla.

Stabilizacja obrotów zachodzi w fazoczułym układzie sprzężenia zwrotnego, przy czym sygnałami odniesienia są impulsy doprowadzane z płytki głównego układu sterowania, generowane przez mikroprocesor. Układ scalony generatora Halla [2]) dokonuje również komutacji między dwoma fazami i generuje impulsy wskaźnikowe. Układ ten, znajdujący się wewnątrz zespołu głowicy, jest bardzo istotny dla poprawności pracy całej pamięci i w przypadku jego uszkodzenia mikroprocesor wysyła odpowiednią informację o błędzie. W celu usunięcia takiej usterki nie trzeba jednak rozmontowywać głowicy, gdyż zapasowy i już odpowiednio przyłączony układ znajduje się w zespole silnika głównego napędu. Aby włączyć go do pracy należy jedynie przeciąć jedno z połączeń znajdujących się na płytce sterowania silnika.

Płytka głównego układu sterowania zawiera obwody kanału odczytu i zapisu, obwody sprzęgające, wyboru głowicy, detekcji błędu i sterowania silnika krokowego. Poszczególnymi operacjami realizowanymi przez tę płytkę steruje znajdujący się na niej mikroprocesor INTEL 8048. Pracuje on w systemie z pamięcią ROM o pojemności 1 K bajtów, pamięcią RAM o pojemności 64 bajtów oraz z 24 liniami wejścia-wyjścia. Częstotliwość sygnału zegarowego wynosi 6 MHz, co powoduje, że czas wykonania rozkazu wynosi 2,5 μs. Najważniejszą funkcją systemu mikroprocesorowego jest sterowanie silnika krokowego, obrazowanie stanu pracy pamięci dyskowej oraz sygnalizacja błędu.

Po włączeniu zasilania mikroprocesor testuje własną pamięć, a następnie sprawdza obecność impulsów wskaźnikowych, generowanych w układzie silnika napędowego. Następnie przeprowadza kontrolę prędkości obrotowej, co trwa 25 s. Po sprawdzeniu właściwej pracy silnika, mikroprocesor rozpoczyna kalibrację pozycjonera, ustawiając go na ścieżkę 00 i sprawdzając, czy pozycjonery i detektor

[2]) Generator Halla, zwany też hallotronem, jest przyrządem półprzewodnikowym, wykorzystującym zjawisko Halla, które polega na pojawianiu się w półprzewodniku dodatkowego pola elektrycznego pod wpływem pola magnetycznego

ścieżki zerowej pracują poprawnie. Potem ponownie sprawdza się prędkość obrotową silnika, co kończy test. Jeśli wszystkie sprawdzenia wypadły pomyślnie, to generowany jest sygnał „zespół napędowy gotów". W przypadku, gdy jakiś sprawdzian wypadł negatywnie, na płycie czołowej zespołu napędowego jest wyświetlany kod błędu, a mikroprocesor przerywa swą pracę. Rodzaj błędu (w kodzie czterobitowym) jest określony przez kombinację diod świecących. Poprawność pracy jest sygnalizowana przez cały czas działania pamięci (np. gdy procesor czeka na impulsy krokowe z układu sprzęgającego, wyświetla się wartość prędkości obrotowej silnika).

* * *

Niezmiernie istotny wpływ na wartości podstawowych parametrów pamięci (jak: pojemność, czas dostępu, niezawodność pracy) ma mechanizm pozycjonera. W wielu pamięciach dyskowych zastosowano wyłącznie mechaniczne rozwiązania; w opisywanej pamięci natomiast ustawianiem (przesuwaniem i hamowaniem) pozycjonera steruje mikroprocesor; skutkiem czego uzyskano wyjątkowo płynną i precyzyjną pracę tego podzespołu. Dodatkową zaletę zastosowanego rozwiązania stanowi fakt, iż histereza silnika krokowego nie ma wpływu na jakość pracy pamięci — głowica dochodzi do ścieżek zawsze od tej samej strony.

Warto podać dwa bardzo istotne parametry użytkowe pamięci dyskowych typu Winchester produkcji firmy RODINE LTD: średni czas potrzebny na przeprowadzenie naprawy — 30 minut i średni czas eksploatacji pamięci — 12 tys. godzin.

LITERATURA

[1] Dudson M.: Winchester Technology Brings More Storage to Microsystems. Computer Design, September 1982
[2] Kirby M.: Selection Criteria Ease Choice between 8- and 14-inch Winchester Disc Drives. Computer Design, October 1980
[3] Niżankowski A.: Pamięci dyskowe typu Winchester z niewymiennym pakietem dysków. Biuletyn Techniczny MERA, nr 3, str. 21—24, 1981.

Opracowała MARIA WAŚNIEWSKA

ALGORYTMY

BASIC

Generowanie liczb pseudolosowych

W praktyce programowania często spotkać się można z sytuacjami, w których nie wystarcza deterministyczny model obliczeń. Przykładem może być tu symulacja cyfrowa, w której wiele czynników wpływających na przebieg rozważanych procesów można opisać jedynie w sposób statystyczny. Szeroką klasę metod związanych z generowaniem liczb pseudolosowych stanowią też algorytmy probabilistyczne.

W ostatnich artykułach tej rubryki zaprezentowaliśmy kilka takich algorytmów. Przedstawione poniżej procedury pomyślane są jako ich uzupełnienie. Prostota generatorów liczb pseudolosowych skłoniła nas do użycia notacji BASICOWEJ (z wyjątkiem pierwszej funkcji), co chyba spotka się z przychylnym przyjęciem użytkowników systemów mikroprocesorowych. Przy większości procedur podajemy szkicowy opis metod, odsyłając zainteresowanych ich matematyczną analizą do drugiego tomu monografii D.E. Knutha „The Art of Computer Programming" lub do dostępnej na naszym rynku książki R. Zielińskiego „Generatory liczb losowych" (WNT, 1979).

Opis funkcji RAND

(por. Forsythe G.E., Malcolm M.A., Moler C.B.: Computer Methods for Mathematical Computations). Prentice-Hall, Inc., 1977)

Wartościami funkcji RAND są kolejne rzeczywiste liczby pseudolosowe z przedziału (0,1) generowane z rozkładem równomiernym. Jedynym parametrem RAND jest zmienna całkowita IY. Program wywołujący RAND powinien zainicjować tę zmienną, nadając jej wartość dowolnej liczby całkowitej. Pomiędzy kolejnymi wywołaniami RAND nie należy zmieniać wartości IY.

RAND korzysta z zewnętrznej funkcji **DSQRT(x)**, obliczającej pierwiastek z **x** (wynik oraz **x** są liczbami podwójnej precyzji) oraz z funkcji **IDINT** (x), której wynikiem jest część całkowita x. Użyjemy również symbolicznego zapisu π na oznaczenie znanej stałej matematycznej (π = 3.14...). Zmienne podwójnej precyzji oraz DSQRT i IDINT można z powodzeniem zastąpić przez zmienne rzeczywiste (oraz właściwe funkcje).

W zapisie procedury funkcyjnej RAND odstąpimy od notacji BASICOWEJ na rzecz FORTRANU. Wynika to z faktu, że wykorzystywane są w tym przypadku własności maszynowej reprezentacji liczb całkowitych (a więc nie dające się zapisać w standardowym BASICU). Przetłumaczenie zapisu FORTRANOWEGO na program BASICOWY z odpowiednimi wstawkami w języku wewnętrznym maszyny pozostawiamy czytelnikom.

```
      REAL FUNCTION RAND(IY)
      INTEGER IY,IA,IC,M2,M,MIC,IDINT
      REAL S
      DOUBLE PRECISION HALFM, DSQRT
      DATA M2/0/
      IG(M2.NE.0) GOTO 20
C JEŚLI TO JEST PIERWSZE WYWOŁANIE, TO WYLICZAMY
                                        DŁUGOŚĆ
C SŁOWA MASZYNOWEGO I WARTOŚCI POCZĄTKOWE ODPO-
                                        WIEDNICH
C ZMIENNYCH
      M = 1
10    M2 = M
      M = 2 * M2
      IF(M.GT.M2) GOTO 10
      HALFM = M2
      IA = 8*IDINT(HALFM*π/32.0)+5
      IC = 2*IDINT(HALFM*/0.5 — DSQRT(3.0)/6.0))+1
      MIC = (M2—IC)+M2
      S = 0.5/HALFM
C TERAZ WYLICZENIE KOLEJNEJ LICZBY PSEUDOLOSOWEJ
20    IY = IY*IA
      IF(IY.GT.MIC) IY = (IY—M2)—M2
      IY = IY+IC
      IF(IY/2.GT.M2) IY = (IY—M2)—M2
      IF(IY.LT.O) IY = (IY+M2)+M2
```

```
RAND = FLOAT(IY)*S
RETURN
END
```

We wszystkich następnych procedurach przez symboliczny zapis:

etykieta X = kolejna liczba losowa

rozumieć będziemy wywołanie jakiegokolwiek generatora liczb losowych o rozkładzie równomiernym w przedziale (0,1) — może nim być np. opisana poprzednio funkcja RAND.

Opis funkcji RNORM

Funkcja RNORM generuje kolejne liczby pseudolosowe z rozkładem normalnym o wartości oczekiwanej S i odchyleniu standardowym W. S i W są zmiennymi globalnymi. Wynik przechowywany jest na zmiennej R.

Zasada działania funkcji RNORM opiera się na centralnym twierdzeniu granicznym, z którego wynika, że rozkład sumy niezależnych liczb losowych o rozkładzie równomiernym dąży do rozkładu normalnego. W poniższej procedurze zgodnie z „prawem tuzina" zmienną losową o rozkładzie normalnym przybliżamy za pomocą sumy dwunastu liczb pseudolosowych generowanych z rozkładem równomiernym na przedziale (0,1).

```
100 REM   S — ŻĄDANA WARTOŚĆ OCZEKIWANA
110 REM   W — ODCHYLENIE STANDARDOWE, R — WYNIK
120 R = 0
130 FOR I=1 TO 12
140 X = kolejna liczba losowa
150 R = R+X
160 NEXT I
170 R = (R—6)*W+S
180 RETURN
```

W kolejnych funkcjach korzystać będziemy z metody odwracania dystrybuanty (p. wspomniana książka R. Zielińskiego). Jej podstawę stanowi spostrzeżenie, że jeśli R jest zmienną losową o rozkładzie równomiernym, zaś funkcja F spełnia warunki:

- $F(-\infty) = 0$, $F(+\infty) = 1$
- F jest ściśle rosnąca i ciągła,

to zmienna losowa $X = F^{-1}(R)$ ma rozkład o dystrybuancie $F(x)$.

Opis funkcji REXP

Rozpatrzmy zmienną losową W o rozkładzie wykładniczym i wartości oczekiwanej równej S (S>0). Dystrybuanta tej zmiennej ma postać:

$$F(x) = \begin{cases} 1 - e^{-x/S} & \text{dla } x \geq 0 \\ 0 & \text{dla } x < 0 \end{cases}$$

Stąd:

$$F^{-1}(x) = \begin{cases} -S*\ln(1-x) & \text{dla } x < 1 \\ 0 & \text{dla } x \geq 1 \end{cases}$$

Ponieważ zmienną x generujemy losowo z rozkładem równomiernym na przedziale (0,1), powyższe wyrażenie można uprościć do:

$$F^{-1}(x) = -S*\ln(x).$$

Uzyskujemy zatem procedurę REXP, która wykorzystuje standardową funkcję LN(X) obliczając logarytm naturalny z X:

ZAMÓWIENIE

Zamawiamy prenumeratę (kwartalną, półroczną, roczną, roczną-stałą)* od dnia ..... 198... r. niżej wymienionych czasopism wydawanych i kolportowanych przez Wydawnictwo NOT SIGMA **:

lp.	Tytuł	Liczba etgz.	Uwagi
1.			
2.			
3.			
...			

Zamówione czasopisma prosimy wysyłać na podany niżej adres:

Nazwa instytucji: . . . . . .

. . . . . .

ul. . . . . nr . . . .

. . . . . .

miejscowość

Główny Księgowy ..... Dyrektor

UWAGA:

1. Zamówienia prosimy wysyłać w jednym egzemplarzu.
2. W przypadku kilku różnych odbiorców prasy, do zamówienia należy dołączyć wykazy czasopism z dokładnym adresem odbiorców — osobno dla każdego odbiorcy.
3. Zakład Kolportażu nie wystawia rachunków; w związku z tym należność za prenumeratę należy przekazać na konto Wydawnictwa: NBP III O/M w Warszawie, nr 1036-7490-139-11 w terminie pięciu dni od wysłania zamówienia. Na przelewie należy koniecznie podać kod pocztowy zamawiającego (płatnika).
4. Bez opłacenia prenumeraty czasopisma nie będą wysyłane.
5. Zamówienia, które nie będą zawierać powyższych danych, będą zwracane nadawcom — na ich koszt.
6. Informacji o prenumeracie udziela Zakład Kolportażu: tel. centr. 40-00-21 wew. 293, 295; bezpośredni 40-35-89.

* niepotrzebne skreślić

** pełny wykaz tych czasopism zamieściliśmy w numerach 4-5, 6 i 7/1982 — przyp. red.

```
200 REM S — WARTOŚĆ OCZEKIWANA, R — WYNIK
210 X = kolejna liczba losowa
220 R = —S*LN(X)
230 RETURN
```

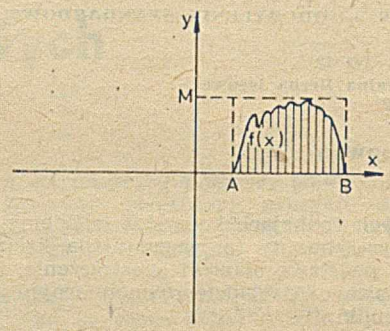
Opis funkcji RPOW

Załóżmy teraz, że zmienna losowa **W** ma rozkład potęgowy z parametrem **A** ($A > 0$), tzn. dystrybuanta wyraża się wzorem:

$$F_A(x) = \begin{cases} 0 & \text{dla} \quad x < 0 \\ x^A & \text{dla} \quad 0 \leqslant x \leqslant 1 \\ 1 & \text{dla} \quad x > 1 \end{cases}$$

Oczywiście, zgodnie z metodą odwracania dystybuanty zmienna losowa $X = Y^{1/A}$, gdzie **Y** jest zmienną o rozkładzie równomiernym w przedziale (0,1), ma dystrybuantę $F_A(x)$. Stąd uzyskujemy procedurę RPOW, korzystającą ze standardowej funkcji EXP(X), obliczającej e^X oraz z opisanej wcześniej funkcji LN(X):

```
300 REM A — PARAMETR ROZKŁADU, R — WYNIK
310 X = kolejna liczba losowa
320 R = EXP(LN(X)/A)
330 RETURN
```



Funkcja gęstości rozkładu prawdopodobieństwa

Na podobnej zasadzie można uzyskiwać generatory liczb losowych wielu innych rozkładów (np. rozkładu podwójnie wykładniczego czy rozkładu Frécheta). Jeśli zaś dana jest funkcja gęstości prawdopodobieństwa **f(x)**, przy czym $f(A) = f(B) = 0$ i **f** jest ciągła w **(A,B)** (por. rys.), to do generowania odpowiadających jej liczb pseudolosowych z przedziału (A,B) można użyć następującego prostego schematu:

```
400 M - maksimum funkcji f (x) dla x ∈ (A,B)
410 X = kolejna liczba losowa
420 X1 = (B—A)*X+A
430 REM X1 JEST LICZBĄ PSEUDOLOSOWĄ WYGENEROWANĄ
                                    Z ROZKŁADEM
440 REM RÓWNOMIERNYM NA PRZEDZIALE (A,B)
450 X = kolejna liczba losowa
460 X2 = M*X
465 REM LICZBA X1 JEST AKCEPTOWANA, JEŚLI PARA
                                    (X1,X2) LEŻY W OBSZARZE
466 REM ZAKRESKOWANYM NA RYSUNKU
470 IF X2≥f (X1) THEN 410
480 R = X1
490 REM R JEST WYNIKIEM
500 RETURN
```

ANDRZEJ SZAŁAS
ZBIGNIEW ŚWIRSKI

miejscowość _____ dnia _____

zamawiający — nazwa

ulica, numer

kod pocztowy _____ miejscowość

konto bankowe: _____

WYDAWNICTWO NOT SIGMA
Zakład Kolportażu
00-950 Warszawa
skr. poczt. 1004

Informatyka w handlu

Informatyka pojawiła się w polskim handlu już w 1965 roku, kiedy w Wojewódzkim Przedsiębiorstwie Handlu Odzieżą w Warszawie uruchomiono pierwszy system informatyczny. Kilka lat później, w Instytucie Handlu Wewnętrznego i Usług podjęto systematyczną obserwację i analizę rozwoju informatyki w handlu wewnętrznym — państwowym i spółdzielczym. Analiza ta miała na celu dostarczenie centralom handlowym materiałów ułatwiających kierowanie rozwojem informatyki.

Obserwację i analizę rozwoju informatyki w handlu wewnętrznym prowadzono w podziale na trzy umowne etapy. Analiza dwóch pierwszych została przez Instytut opublikowana [1]); analiza trzeciego — została ostatnio ukończona.

W pierwszym etapie (1965—1970) badaniami ankietowymi objęto wszystkie jednostki handlu państwowego i spółdzielczego, które stosowały systemy zmechanizowanego lub zautomatyzowanego przetwarzania danych co najmniej przez jeden rok (przy braku własnych komputerów w resorcie). Oprócz szerszej charakterystyki tych jednostek, opracowano wówczas także wnioski dla central handlowych. Okazało się m.in., że jednostki terenowe zdane są głównie na własną inicjatywę, że ze strony central otrzymują niedostateczną pomoc metodyczną, organizacyjną i szkoleniową. Wynikał stąd postulat znacznego zwiększenia w następnych latach zakresu działalności koordynacyjnej central, zaś jednym z jej elementów miała być daleko posunięta centralizacja decyzji związanych z informatyką w resorcie.

Etap drugi trwał właściwie do połowy 1976, kiedy to wprowadzono generalne zmiany w organizacji handlu wewnętrznego. W tym okresie nie było już możliwe objęcie badaniami wszystkich, licznych użytkowników systemów. W porozumieniu z centralami wytypowano więc 61 jednostek handlowych, w których system funkcjonował co najmniej jeden rok (oznaczało to ok. 30% systemów funkcjonujących w handlu państwowym oraz 100% systemów w handlu spółdzielczym).

Po 1970 roku jednostki handlowe zaczęły korzystć z usług nielicznych jeszcze ośrodków w resorcie: Ośrodka Elektronicznego Przetwarzania Danych HW (głównie jednostki handlu państwowego), Zakładu Przetwarzania Danych przy WZSR „Samopomoc Chłopska" w Szczecinie oraz Ośrodka Zmechanizowanego Obrachunku przy WSS w Krakowie, a także z usług ZETO. Użycie cudzego sprzętu okazało się zresztą rozwiązaniem najbardziej korzystnym ekonomicznie.

Nastąpiła przy tym znaczna aktywizacja czynności koordynacyjnych i instruktażowych ze strony central handlowych, która w połączeniu z ujednoliceniem sprzętu w ośrodkach obliczeniowych pozwoliła na coraz szersze stosowania systemów typowych.

W połowie 1976 dokonano podziału obsługi rynku między trzy wielkie organizacje: nową Centralę Państwowego Handlu Wewnętrznego, skupiającą obrót wszystkimi podstawowymi artykułami przemysłowymi na rynku miejskim, CZSS „Społem", obejmujący miejski handel w zakresie artykułów spożywczych oraz drobnych artykułów gospodarstwa domowego oraz CZSR „Samopomoc Chłopska", odpowiedzialny za zaopatrzenie rynku wiejskiego w artykuły przemysłowe i spożywcze. Reorganizacja handlu spowodowała zmiany w przynależności obiektów handlowych (m.in. magazynów), w których funkcjonowały już systemy informatyczne, przepływ specjalistów-informatyków, inne usytuowanie komórek informatycznych itd. W sumie można stwierdzić, że konsekwencje tych zmian miały największy wpływ w branży spożywczej, gdzie pierwsze wdrożenia informatyki przesunięto o rok.

Analiza trzeciego etapu, obejmującego lata 1975—1982, nie mogła już opierać się na badaniach ankietowych, zarówno z uwagi na zbyt dużą liczbę funkcjonujących systemów, jak i rozpoczęcie kolejnej reorganizacji aparatu handlu, wyrażającej się m.in. w likwidacji Centrali Państwowego Handlu Wewnętrznego. Z tego powodu a-

[1]) Jerczyńska M.: Informatyka w handlu wewnętrznym. I etap rozwoju 1965—1970. Biblioteka IHW nr 84, Warszawa, 1972; Jerczyńska M.: Systemy informatyczne w handlu wewnętrznym. II etap rozwoju 1970—1975, Biblioteka IHWiU nr 160, Warszawa, 1977

GIEŁDA INFORMACJI

Publikujemy bezpłatnie:

- krótkie prezentacje nowych, nie wykorzystanych dostatecznie systemów
- doniesienia o innowacjach programowych
- ogłoszenia dotyczące nowych rozwiązań technicznych sprzętu informatycznego
- indywidualne oferty informatyków poszukujących pracy oraz informacje o wolnych etatach
- ogłoszenia w sprawie szkoleń informatycznych
- inne informacje ułatwiajace doraźną organizację informacji w Polsce.

TELEKOMUNIKACJA I KOMPUTER

W ramach obecnego Światowego Roku Telekomunikacji, proklamowanego przez Zgromadzenie Ogólne ONZ, Muzeum Poczty i Telekomunikacji we Wrocławiu — przy współudziale ZETO Wrocław — zorganizowało wystawę TELEKOMUNIKACJA I KOMPUTER. Wystawa trwa od maja do 10 września. W jej ramach zaprezentowano cykl fotogramów KOMPUTER W SŁUŻBIE POCZTY I TELEKOMUNIKACJI. W sali Muzeum zainstalowano minikomputer MERA-100, na którego monitorze wyświetlana jest specjalnie przygotowana „gazetka".

Adres Muzeum Poczty i Telekomunikacji: 50-954 Wrocław, ul. Krasińskiego 1, I p.

KELNER

ZETO Wrocław oferuje System Rozliczeń Kelnerskich KELNER, przeznaczony dla hoteli oraz dużych zakładów gastronomicznych z liczną obsługą kelnerską. Ma on na celu rozliczanie utargów kelnerskich w różnych przekrojach. Oprogramowanie: system operacyjny FDOS oraz język BASCOF. Minimalny zestaw sprzętu: minikomputer MERA-100-3312 z pamięcią operacyjną 32 KB, dysk elastyczny PL X 45 D (4 jednostki), drukarka DZM-180, pamięć magnetyczna PK-1 i monitor ekranowy. System może być wdrażany w zakładach nie posiadających własnego sprzętu, a mających jedynie możliwość jego dzierżawienia.

System ten umożliwia tworzenie raportu kasowego sprzedaży gastronomicznej i zbioru miesięcznych utargów netto i brutto dla punktów obsługi oraz emituje zestawienia wynikowe wykorzystywane przy naliczaniu wynagrodzeń i tworzeniu sprawozdań.

Zgłoszenia przyjmuje: kierownik Działu Marketingu, mgr Tadeusz Czerniewski; ZETO Wrocław, ul. Ofiar Oświęcimskich 7/13; 50-069 WROCŁAW; tel. centr. 44-54-31 do 37; teleks 0712533 ZETO PL; konto bank. 93044-1300 IV OM WROCŁAW.

nalizę oparto na materiałach zgromadzonych w centralach oraz systematycznych obserwacjach własnych, wspieranych kilkoma badaniami cząstkowymi.

Wśród cech o podstawowym znaczeniu należy wymienić:

• **Znaczny wzrost centralizacji programowania, planowania i kierowania informatyką w centralach handlowych.** Komórki informatyczne central nie tylko opracowywały programy i plany rozwoju, ale dysponowały także środkami (m.in. Funduszem Postępu Technicznego) zapewniającymi ich realizację. Wykonywały one ponadto zadania koncepcyjne, organizatorskie i instruktażowe. Zaczęły być więc spełniane postulaty oddolne, zgłaszane w latach wcześniejszych, domagające się zwiększenia form pomocy za pośrednictwem centralnej koordynacji. Doświadczenia tego etapu dowiodły, że centralizacja podstawowych decyzji (np. w zakresie kodów) i działań była najbardziej racjonalnym sposobem rozwoju informatyki w handlu.

• **Centralne sterowanie rozwojem informatyki,** które spowodowało, że zgodnie z planami systemy informatyczne zaczęły obejmować także wyższe szczeble zarządzania obrotem towarowym. Wprawdzie nie objęły one jeszcze szczebla resortu, jednak zaczęto już eksploatować stopniowo rozbudowywane systemy obsługi różnych agend, w tym — kierownictwa central. Większa rozbudowa tych systemów, także opartych na zbiorach danych z systemów obiektowych, przewidywana była na lata osiemdziesiąte. Jednocześnie prowadzono rozbudowę systemów obiektowych na szczeblu wojewódzkim, dążąc do uzyskania kompleksowych, wielodziedzinowych systemów informatycznych przeznaczonych dla przedsiębiorstw, spółdzielni czy związków spółdzielni. Wszystkie one wykazywały jeden podstawowy kierunek: magazynowy obrót towarowy (ogniwo hurtu). Zastosowania w zakresie handlu detalicznego planowano na lata późniejsze, wiążąc to z możliwościami zastosowania terminali kasowych produkcji polskiej, jaką podjęto na początku lat osiemdziesiątych w Centrum Komputeryzacji Rynku CEKAR (dawny OEPD HW).

• **Przyjęcie strategii wdrażania systemów typowych:**

— w CPHW — jednego uniwersalnego systemu wielobranżowego dla wszystkich kompleksów towarów nieżywnościowych

— w „Społem" — jednego systemu dla wszystkich branż artykułów spożywczych w oparciu o adaptację istniejących rozwiązań; natomiast dla artykułów przemysłowych — nowego systemu dla artykułów gospodarstwa domowego, a w dalszej perspektywie

— jednego uniwersalnego systemu dla wszystkich branż.

Jednolitość systemów informatycznych wewnątrz każdej organizacji handlowej uzyskiwano poprzez stosowanie jednolitych kodów (towarów, zaszłości gospodarczych), dokumentów źródłowych oraz rozwiązań projektowo-programistycznych.

Systemy typowe w handlu państwowym dotyczyły magazynów przedsiębiorstw, w spółdzielczym handlu miejskim — spółdzielni oraz oddziałów, zaś w spółdzielczym handlu wiejskim — wojewódzkich związków oraz ich zakładów.

• **Wykształcenie się specyficznego modelu rozwiązań techniczno-organizacyjnych.** Polegał on na tym, że wszystkie przedsiębiorstwa eksploatowały systemy informatyczne wyłącznie w oparciu o sprzęt oraz usługi projektowo-programistyczne jednostek zewnętrznych (ZETO, zakłady terenowe CEKAR oraz Zakład Przetwarzania Danych „Samopomoc Chłopska" w Szczecinie). Korzystano głównie z komputerów serii ODRA 1300 oraz RIAD, a także szczególnie użytecznych w handlu minikomputerów MERA 9150.

• **Rozwój prac koncepcyjnych dotyczących programowania całości rozwoju informatyki w resorcie.** Prace te rozpoczęto w Ośrodku EPD HWiU w 1976 po wspomnianej reorganizacji aparatu handlu, przedstawiając początkowo program z przewagą elementów technicznych. Pierwszy program o przewadze elementów ekonomicznych opracowano w CEKAR w 1980 r. Stanowił on kompleksową koncepcję systemu resortowego, obejmującego całą działalność rynkową. Po raz pierwszy omówiony został w nim problem centralnej bazy danych przeznaczonej na potrzeby systemu resortowego. Równolegle w IHWiU prowadzono odpowiednie prace naukowo-badawcze. Stanowiły one teoretyczne wsparcie dla prac koncepcyjnych, podejmowanych w samym resorcie i w ośrodkach obliczeniowych.

* * *

Jak wynika z przedstawionej analizy, reorganizacja aparatu handlu w latach 1975—1976 spowodowała opóźnienie wdrożeń systemów informatycznych. Okres ich weryfikacji, adaptacji lub projektowania od nowa, powodujący zresztą nowe koszty, trwał 2—3 lata. Od 1978 roku wdrażano głównie systemy typowe, nakierowane na tworzenie podstaw dla przyszłego kompleksowego systemu informacji o rynku wewnętrznym. Znacznie większa „czystość branżowa" organizacji handlowych była czynnikiem sprzyjającym masowemu wdrażaniu systemów typowych. Dobrym przykładem jest tu skupienie handlu artykułami spożywczymi na terenie miast w gestii „Społem". Można powiedzieć, że interesy informatyki były w tym przypadku zgodne z intencją przeprowadzanej reorganizacji — choć prawdopodobnie nie zawsze była ona zgodna z interesami konsumentów. Ten sprzyjający układ organizacyjny handlu został jednak w części zlikwidowany w roku 1982; można więc stwierdzić, że rozpoczął się nowy etap rozwoju informatyki w handlu.

Etap ten będzie zależny od wymogów reformy gospodarczej i stanu nierównowagi rynkowej; te dwa fakty będą rozstrzygać o tempie i kierunkach dalszego rozwoju informatyki w handlu.

Co można ocalić z osiągnięć dotychczasowych? Spróbujmy odpowiedzieć.

Systemy obiektowe miały głównie charakter ewidencyjno-kontrolny, dostosowany do potrzeb ówczesnego rynku, na którym nie występowała równowaga i nie było potrzeby badania popytu. Prawdopodobnie ten właśnie fakt spowodował, że w latach 1980—1982 przedsiębiorstwa nie rezygnowały na ogół z eksploatowanych systemów. Systemy obiektowe będą się zapewne dalej rozwijać (choć początkowo w zwolnionym tempie), niemniej inny będzie ich charakter. Jak wynika z doświadczeń roku 1982 i z założeń reformy gospodarczej, będą one z pewnością rozbudowywane w dziedzinie analizy finansowej, cen oraz innych elementów wpływających na ostateczny wynik działalności gospodarczej. Będą one zapewne nadal miały charakter typowy — są to bowiem niewątpliwie systemy najtańsze.

Realizacja systemów wyższego rzędu, w tym resortowego, napotyka na znacznie większe trudności. Jednolitość systemów informatycznych w ramach każdej organizacji handlowej uzyskiwano m.in. poprzez stosowanie jednolitych systemów kodowania. Zapewniało to spójność wszystkich systemów wewnątrz każdej organizacji i na ówczesnym etapie rozwoju było wystarczające. Wymagało to jednak jednoczesnego podjęcia działań koordynacyjno-unifikacyjnych w skali resortu, które pozwoliłyby na etapowe przechodzenie do stosowania jednolitych kodów i stopniowej agregacji zdecentralizowanych zbiorów danych w całym obrocie towarowym. Prace takie, uwzględniające stosowane kody, nie zostały jednak podjęte. Niemniej ujednolicenie bazy normatywnej, w tym przede wszystkim indeksu towarowego, jest podstawowym warunkiem wykorzystania zbiorów informacji rynkowych, gromadzonych obecnie w sposób zdecentralizowany. Być może problem ten mógłby być rozwiązany poprzez wprowadzenie jednolitego kodu materiałowo-towarowego, przygotowywanego dla systemu SPIS.

Drugą podstawową trudnością jest sprawa korzystania ze zbiorów informacji rynkowych, przechowywanych w sposób zdecentralizowany. W każ-

dej sytuacji rynkowej informacje o działalności przedsiębiorstw handlowych oraz o zaopatrzeniu rynku są niezbędne dla wyższych szczebli zarządzania czy koordynacji rynku wewnętrznego. Jednym ze sposobów rozwiązania tego problemu mogłoby być porozumienie resortu (lub innej instytucji) z ośrodkami obliczeniowymi, eksploatującymi systemy informatyczne jednostek handlowych (systemy obiektowe). Stosowanie w tym przypadku jednolitych kodów umożliwiłoby agregację informacji rynkowych z systemów obiektowych. W ten sposób decentralizacja działalności informatycznej i przechowywania zbiorów danych nie stanowiłyby przeszkody w ich wykorzystaniu przez instytucje centralne. Że z organizacyjnego i technicznego punktu widzenia jest to możliwe, świadczą doświadczenia Czechosłowacji i NRD. Aby jednak w aktualnych i przyszłych warunkach rynkowych taki system informacji o rynku mógł funkcjonować, niezbędne jest opracowanie programu rozwoju informatyki w skali resortu. Wbrew pozorom nie oznacza to nawoływania do dawnych form zarządzania, bo warunki gospodarcze po prostu by na to nie pozwoliły. Oznacza to natomiast, że w programie takim powinny być określone potrzeby informacyjne resortu oraz formy ich realizacji. Formy te zaś powinny obejmować zarówno korzystanie ze zbiorów systemów obiektowych, jak i np. z badań panelowych. Decentralizacja informatyki nie musi oznaczać braku informacji, wymaga jednak znacznie większych wysiłków, by ją zgromadzić.

MARIA JERCZYŃSKA
Instytut Rynku
Wewnętrznego i Konsumpcji
Warszawa

Lawina komputerów domowych

Nawet najbardziej optymistyczne prognozy, obejmujące ubiegły rok, nie zdołały przewidzieć tak szaleńczego wzrostu sprzedaży komputerów domowych (ang. home computer). W ostatnim kwartale sprzedaż ta stanowiła prawie połowę całorocznego obrotu. Oczywiście, tak chłonny rynek spowodował maksymalne wykorzystanie wszytkich mocy produkcyjnych. Choćby w firmie TEXAS INSTRUMENTS linia wytwarzająca model 99/4a pracuje 24 godziny na dobę przez siedem dni w tygodniu. Na początku 1982 roku produkowano w USA miesięcznie ok. 10 tys. tych komputerów. W październiku i listopadzie po 100 tys., na grudzień przewidziano osiągnięcie poziomu 200 tys. (sezon prezentów gwiazdkowych).

W dniach 6—7 grudnia ub.r. w Dallas (Texas) odbyło się forum poświęcone mikrokomputerom. Z tej okazji firma FUTURE COMPUTING INC. (USA), zajmująca się badaniem handlu detalicznego komputerami, przygotowała ciekawe dane dotyczące tego rynku. Szczególnie interesujący jest dział komputerów domowych (w cenie do 500 $), pracujących w oparciu o oprogramowanie kupowane w postaci gotowych kostek pamięci ROM. W tej dziedzinie zrealizowano 24% rocznego obrotu na rynku mikrokomputerów, który w Stanach Zjednoczonych szacowano na 1,145 mld $ (z czego 860 mln stanowił sprzęt). Liderami w tym wyścigu były firmy: COMMODORE BUSINESS MACHINES, TEXAS INSTRUMENTS, ATARI i TANDY (seria RADIO SHACK). Wyniki tych firm przedstawiono w tabeli (TIMEX i CINCLAIR stanowią odrębną kategorię, gdyż obejmują mikrokomputery w cenie poniżej 100 dolarów). W sumie sprzedaż w 1982 r. była ok. 6-krotnie większa niż w roku poprzednim. Natomiast na bieżący rok przewiduje się obroty w wysokości 1,7 mld $ w handlu sprzętem oraz 680 mln $ w handlu oprogramowaniem.

Pani Portia Isaacson, prezes FUTURE COMPUTING, tak komentuje te rezultaty: „Rynek po prostu oszalał. Wskaźniki mówią, że obecnie sprzedane będzie wszystko, co znajdzie się na półkach sklepowych. To jest jeden z tych przypadków, gdy ma się przed sobą dane, wie się, że są one prawdziwe, ale ciągle nie można w nie uwierzyć".

FUTURE COMPUTING łączy ten nagły wzrost z kilkoma przyczynami: wyraźną obniżką cen podstawowych jednostek centralnych, kampanią reklamową w telewizji i większym zainteresowaniem ze strony prasy codziennej. Fenomen 1982 roku w dziedzinie domowych komputerów można porównać do fenomenu zainteresowania grami telewizyjnymi (i jest formą naturalnej jego kontynuacji) dwa lata wcześniej — z tą tylko różnicą, że wówczas była to krótkotrwała eksplozja, obecnie zaś można sądzić, iż nie jest to sprawa jednego sezonu.

Sprzedaż komputerów domowych firm amerykańskich (w tys. sztuk)

Producent (model)	Lata		
	1981	1982	
		w USA	ogółem
COMMODORE (VIC-20)	90	500	700
TEXAS INSTRUMENTS (99/4a)	40	490	570
ATARI (400/800)	110	430	500
RADIO SHACK (TRS-80 COLOR COMPUTER)	50	250	300
TIMEX	—	590	590
SINCLAIR (ZX-81)	—	200	520

Ten nagły wzrost sprzedaży oraz przywiązanie użytkownika do jednego źródła oprogramowania (kostki ROM) i urządzeń dodatkowych (czyli do wytwórcy sprzętu podstawowego) musi wyłonić już wkrótce pewne standardy. Pani Isaacson przewiduje, że standardy systemów operacyjnych i oprogramowania dla najmniejszych maszyn cyfrowych pojawią się już w bieżącym roku. Spodziewa się ona „wymuszenia" jednolitej architektury przez jednego lub dwóch z obecnie czołowych producentów — na innych firmach. Chyba że nagle pojawi się na rynku tani, oparty na pamięci ROM, komputer firmy APPLE albo IBM...

na podstawie
„Electronics" nr 24/82
Oprac. MAREK SOBCZYK

Rynek 1982

Mikroprocesory i pamięci

Liczbę sprzedanych w 1982 roku mikroprocesorów szacuje się na 250 mln sztuk, co oznacza 25-procentowy wzrost w stosunku do poprzedniego roku. Pojemność wyprodukowanych pamięci zwiększyła się natomiast podwójnie: 13 bilionów bitów w 1982 w stosunku do 6,4 w 1981. Zdecydowaną większość tego rynku stanowią pamięci dynamiczne (70%) i one decydują o pozycji producentów.

MIKROPROCESORY

Na rynku mikroprocesorów można zaobserwować dwie różne tendencje: z jednej strony stabilizację produkcji mikrokomputerów 4-bitowych, z drugiej — dynamiczny rozwój mikroprocesorów 16-bitowych. Nazwą mikrokomputerów 4- i 8-bitowych określa się tutaj układy, które na jednej płytce krzemu (a więc w jednym układzie scalonym) jednoczą mikroprocesor, 64 lub 128 bajtów pamięci RAM, 1 do 2 K bajtów pamięci ROM oraz ok. 20 wejść/wyjść.

Mikrokomputery 4-bitowe

Mikrokomputery te stosowano początkowo do wprowadzenia cyfr z klawiatury. Dzisiaj wykorzystuje się je szeroko w telewizorach, maszynach do zmywania, magnetowidach, piekarnikach itp. Takie zastosowania wymagają bezwzględnego przestrzegania określonych przez użytkownika warunków. W praktyce oznacza to, że nawet w przypadku awarii zasilania konieczne jest zarejestrowanie emisji na magnetowidzie czy włączenie piekarnika o określonej godzinie. Wraz z pojawieniem się mikrokomputerów 8-bitowych zmniejszyła się atrakcyjność 4-bitowych, ale ich udział rynkowy wynosi nadal 56,78% (ok. 140 mln egzemplarzy). Przewiduje się, że w 1985 roku wyprodukuje się 200 mln tego typu mikrokomputerów, co oznacza wprawdzie wolny, ale ciągły wzrost na poziomie ok. 10% rocznie.

Rynek układów 4-bitowych zdominowały dwie firmy: NEC, która wyprodukowała 36 mln egzemplarzy (26% całej produkcji), a TEXAS INSTRUMENTS 32 mln (22%). W następnej kolejności znajdują się firmy: FUJITSU — 15 mln, NATIONAL SEMICONDUCTOR — 14 mln oraz HITACHI — 12 mln (wszystkich razem wyprodukowano 140 mln). Średnia cena mikrokomputera 4-bitowego wyniosła ok. 2 $.

Mikrokomputery 8-bitowe

W ubiegłym roku dostarczono na rynek ponad 50 mln egzemplarzy tych mikrokomputerów (20% ogólnej liczby mikrokomputerów). Średnia cena jednego egzemplarza wyniosła 3,6 $.

Do chwili obecnej ten typ mikrokomputera wytwarzano wyłącznie technologią NMOS i stosowano w małych automatach, wagach elektronicznych, urządzeniach peryferyjnych itp. Pojawienie się wersji CMOS pozwoli wprowadzić je do sprzętu powszechnego użytku. Przykładem może być tutaj magnetowid GRUNDIG, wyposażony w mikrokomputer 6805.

Według konstruktorów tych układów, technologia CMOS pozwoliła znacznie zwiększyć moc przetwarzania oraz zapewniła lepszą niezawodność działania (większość ma kilka źródeł zasilania). Na rynku mikrokomputerów 8-bitowych zdecydowanie dominuje seria 8048: 30 mln egzemplarzy, 60% produkcji (6801;6805 — 10 mln, 3870 — 7 mln). Układy będące odpowiednikami serii 8048 firmy INTEL, produkuje dzisiaj ponadto osiem innych firm: AMD, EUROTECHNIQUE, FUJITSU, NATIONAL SEMICONDUCTOR, NEC, PHILIPS, SIEMENS i TOSHIBA.

Mikroprocesory 8-bitowe

W zeszłym roku dostarczono na rynek ok. 50 mln egzemplarzy mikroprocesorów 8-bitowych, co stanowi ok. 20% całego rynku mikroprocesorów. Średnia cena jednego procesora tego typu wyniosła 3 $. Mikroprocesor 8-bitowy znalazł zastosowanie w czterech dziedzinach: urządzeniach peryferyjnych, telekomunikacji, przemyśle i przyrządach pomiarowych.

W urządzeniach peryferyjnych mikroprocesory 8-bitowe stosuje się bardzo szeroko: od prostych maszyn do fakturowania — aż po wyspecjalizowane terminale czy komputery osobiste. W komputerach osobistych dominują dwa układy: 650X (16 mln egz.) i Z-80 (11 mln).

W telekomunikacji mikroprocesor 8-bitowy jest często uzupełnieniem głównego komputera i zajmuje się obsługą transmisji i odbioru. Każda rodzina mikroprocesorów 8-bitowych posiada własne magistrale wyspecjalizowane na potrzeby telekomunikacji: sprzęgi szeregowe asynchroniczne i synchroniczne lub wyspecjalizowane sprzęgi sterowane protokołami HDLC lub SDLC.

Stosowanie mikroprocesora w przyrządach pomiarowych uwarunkowane jest dostępnością wyspecjalizowanej magistrali IEEE 488. Mikroprocesor dołączony do tego typu magistrali kieruje komunikacją pomiędzy różnymi urządzeniami. Może także stanowić część przyrządu pomiarowego i wtedy pośredniczy w kontaktach ze światem zewnętrznym. Wśród układów magistrali IEEE 448 występujących na rynku należy wymienić: HEF 4738 firmy RTC, MC 68488 firmy MOTOROLA, 96 LS 488 firmy FAIRCHILD oraz 8291 firmy INTEL.

W przemyśle mikroprocesor 8-bitowy stosowany jest do sterowania procesami oraz w centralach alarmowych. Zadaniem mikroprocesora jest zbieranie danych dostarczanych przez czujniki. Do tego celu producenci mikroprocesorów proponują również wiele układów towarzyszących. W wielu rafineriach, ciepłowniach czy fabrykach chemicznych, tzn. wszędzie tam, gdzie procesy wymagają stałego nadzoru i regulacji, mikroprocesory wypierają stosowane tu dotychczas minikomputery.

Rynek mikroprocesorów 8-bitowych zdominowały cztery główne rodziny: 650X z 16 mln egzemplarzy (32% całej produkcji), Z-80 z 11 mln (22%), 6800—2/6809 z 10 mln (20%) oraz 8080/8085 z 8 mln (16%).

Mikroprocesory 16-bitowe

Pomiędzy rokiem 1981 a 1982, liczba mikroprocesorów 16-bitowych zwiększyła się na rynku prawie trzykrotnie (2 mln w 1981 i 5,55 mln w 1982). Średnią cenę tego typu mikroprocesora szacuje się na ok. 16 $. Rynek mikroprocesorów 16-bitowych zdominował INTEL 8086/8088 (3 mln egz. — 55% całego rynku). Ważną wciąż pozycję utrzymują nadal procesory pierwszej generacji — CP 1600 firmy GENERAL INSTRUMENT (0,8 mln) oraz 9900 firmy TEXAS INSTRUMENTS (0,6 mln). Natomiast mikroprocesory największej mocy — 68000 i Z-8000 stanowią zaledwie 9% (0,5 mln) i 4,5% (0,25 mln) produkcji. Te ostatnie dwa produkty występują dzisiaj wyłącznie w seriach próbnych. Przewiduje się jednakże, że już w ciągu najbliższych dwóch lat nastąpi pełen rozkwit ich produkcji, w związku z rozwojem sprzętu, w którym są wykorzystane — np. FORTUNE i APPLE LISA, zbudowane w oparciu o 68000, czy OLIVETTI i COMMODORE, w oparciu o Z-8000.

Przejęcie przez IBM 12% kapitału firmy INTEL wzmocni niewątpliwie pozycję mikroprocesora 8086 i jego pochodnych. Niespodzianek należy się spodziewać również ze strony firmy DIGITAL EQUIPMENT.

Ze względu na duże możliwości adresowania (1 M bajt) mikroprocesory 16-bitowe mają ogromny wpływ na szybki rozwój pamięci dynamicznych.

PAMIĘCI

Podobnie jak w przypadku mikroprocesorów, na rynku pamięci można również zaobserwować dwie tendencje: z jednej strony burzliwy rozwój pamięci dynamicznych 64 K, statycznych 16 K i EPROM 64 K, z drugiej — względną stabilizację w produkcji pamięci dynamicznych 16 K, statycznych 4 K i EPROM 16 K (tab.).

Rynek pamięci w 1982

Rodzaj pamięci	Wielkość produkcji (w mln egz.)
16 K dynamiczne	270
64 K dynamiczne	80
4 K statyczne	82
4 K CMOS	30
16 K statyczne	32
16 K EPROM	45
32 K EPROM	40
inne	16
Razem	595

Pamięci dynamiczne

Pamięci dynamiczne stanowią ponad 70% całkowitej pojemności pamięci wyprodukowanych w 1982 roku. Obserwuje się spadek produkcji pamięci RAM 4 K. Ich średnia cena wyniosła aż 2 $ (dla porównania, cena pamięci o pojemności 16 K — 1,75 $). Rynek pamięci 16 K wykazuje tendencje stabilizacji — dominują na nim trzy firmy: MOSTEK, NATIONAL SEMICONDUCTOR i NEC. Największe zapotrzebowanie występuje na pamięci 64 K. Ich produkcja przeżywa prawdziwy rozkwit i w 1982 r. wzrosła aż ośmiokrotnie w stosunku do 1981 r. Rynek tych pamięci opanowany jest niemal zupełnie przez konstruktorów japońskich i tylko jedna firma amerykańska — MOTOROLA — może współzawodniczyć z Japonią.

Pamięci statyczne

Pamięci statyczne stanowią zaledwie 7% całkowitej pojemności pamięci wyprodukowanych w 1982. Na rynku dostępne są dwa rodzaje: szybkie — z czasem dostępu 70 ns oraz wolne — z czasem dostępu rzędu kilkaset ns. Wśród dominujących na rynku pamięci o pojemności 4 K, większą popularnością cieszy się organizacja 1 K × 4 (w odróżnieniu od 4 K × 1). Średnie ceny pamięci statycznych wynoszą: pamięci szybkie — 4 $, pamięci wolne — 2 $, pamięci wykonane w technologii CMOS — 2,5 $.

Pamięci typu EPROM

Pamięci typu EPROM stanowią 20% całego rynku pamięci. Zniknęły prawie z rynku pamięci o pojemności 8 K, natomiast produkcja pamięci o pojemności 16 K i 32 K ustabilizowała się na poziomie 40 mln egzemplarzy w 1982 r. — przy cenie 4 $ za pamięć 16 K i 7 $ za 32 K. Natomiast liczba sprzedanych pamięci o pojemności 64 K zwiększyła się w porównaniu do 1981 r. aż dziesięciokrotnie (średnia cena egzemplarza 14 $).

Oprac. MARIANNA SOBCZYK
na podstawie
01 INFORMATIQUE HERBO z 24.01.1983
i MINIS et MICROS z 31.01.1983

Superkomputery w Hollywood

Przyzwyczailiśmy się, że głównymi odbiorcami superkomputerów są ośrodki badań atomowych czy meteorologicznych. Natomiast mało znany jest fakt stosowania tych maszyn przez firmy zajmujące się produkcją filmów animowanych i trickowych. A powód tych zastosowań jest bardzo prosty — symulacja bardzo złożonych obrazów filmowych, podobnie jak inne zadania symulacyjne, wymaga największych mocy obliczeniowych.

Działająca na tym polu frima DIGITAL PRODUCTIONS INC. już od ponad roku korzysta z superkomputera CRAY 1S/1000. Za jego pomocą zrealizowano film „Tron" Walta Disneya, w którym — jak dotychczas — wykorzystano najpełniej grafikę komputerową.

Stosowane obecnie standardy jakości powodują, że każdy kadr 70-milimetrowego filmu musi być reprezentowany do celów przetwarzania cyfrowego przez matrycę o rozdzielczości 6000×4500 punktów. Jeśli dodać do tego, że każda sekunda filmu składa się z 24 klatek, to nietrudno wyobrazić sobie, że komputerowa produkcja filmów rysunkowych jest obecnie jednym z najbardziej złożonych zadań obliczeniowych. Według przedstawiciela DIGITAL PRODUCTIONS, CRAY 1S jest w stanie „obliczyć" jedną klatkę filmu (1 obraz) w ciągu ok. 15 minut. Pozwala to wykonać w ciągu miesiąca ok. 4 minut gotowego filmu (na minikomputerze typu VAX firmy DEC można by było zrealizować w ciągu roku 2,5 minuty filmu).

Sprawa odpowiedniego sprzętu jest zatem najważniejsza. Dlatego też DIGITAL PRODUCTIONS nabędzie w końcu tego roku pierwszy egzemplarz nowego superkomputera CRAY X-MP/22. Maszyna ta, wyposażona w pamięć operacyjną o pojemności 2 M słów (po 64 bity) i dwa procesory (w każdym 13 jednostek funkcjonalnych, cykl 9,5 ns, prędkość — ponad 400 mln operacji zmiennoprzecinkowych na sekundę) będzie 4—5 razy szybsza od modelu 1S. Powinno to zwiększyć wydajność do 25 minut filmu na miesiąc i jednocześnie pozwolić na realizowanie scen dotychczas nie dających się wykonać — przy dopuszczalnych kosztach.

Przewiduje się także, że model X-MP przyniesie z jednej strony rozszerzenie możliwości artystycznych projektantów filmowych, z drugiej zaś obniżkę kosztów produkcji (cena X-MP jest tylko o ok. 35% wyższa od modelu 1S). Dodajmy, że ten typ obliczeń (przekształcenia trójwymiarowych tablic o milionach elementów) powoduje maksymalne wykorzystanie mocy maszyn firmy CRAY, których architektura nastawiona jest na optymalizację (zrównoleglenie) operacji tablicowych.

Nowa dziedzina zastosowania superkomputerów została powitana z dużym zadowoleniem przez producenta. Ponieważ CRAY RESEARCH wytwarza tylko 7—8 systemów rocznie przy relatywnie wysokiej cenie (11—15 mln $ za model X-MP), sprzedanie dodatkowo jednej czy dwóch maszyn stanowi istotny wzrost. Rzecznik firmy podkreśla także, iż technika generowania filmów przy użyciu superkomputerów może z czasem rozszerzyć swój zakres na filmy naukowe i przemysłowe, a wreszcie na niefilmowe zastosowania wymagające szczególnie realistycznej symulacji.

Oprac. MkS
na podstawie „Electronics"
vol. 55, nr 16 z 11.08.1982

CSNET jednoczy informatyków

Ponad dziesięć lat po uruchomieniu przez ADVANCED RESEARCH PROJECT AGENCY sieci ARPANET, świat naukowy Stanów Zjednoczonych doczekał się własnej sieci komputerowej — CSNET (Computer Scientists NETwork). Jej podstawowym celem jest świadczenie usług sieciowych dla całej społeczności informatyków w USA.

ARPANET łączy obecnie setki komputerów i tysiące ludzi, nie tylko w USA. Użytkownicy korzystają z usług elektronicznej poczty, przesyłają zbiory i uruchamiają zdalnie programy. Jednak sieć ta jest przeznaczona w zasadzie tylko dla tych, którzy pracują na rzecz Departamentu Obrony. Obecnie naukowcy z ośrodków uniwersyteckich otrzymają podobne narzędzie. CSNET została założona w 1982 przez 40 organizacji, ale już wkrótce pojawiło się 70 następnych zgłoszeń. Oprócz uniwersytetów (np. w Wisconsin, Purdue, Delaware, Kalifornia, Princenton, Waszyngtonie, Utah), sieć połączy także placówki naukowe w przemyśle: Bell Laboratories, Rand Corp., HP Laboratories, laboratorium badawcze IBM Corp., Tektronix i ośrodek badań nad sztuczną inteligencją w Fairchild Semiconductor.

CSNET ma łamać bariery, jakie stworzyły różne techniki sieciowe, musiano więc stworzyć specjalne oprogramowanie. Jest ono na tyle uniwersalne, że może zostać użyte nie tylko przez informatyków, ale i przez inne grupy zawodowe, które potrzebują własnej sieci. Jeśli — na przykład w przyszłości powstanie „Chemnet" czy „Physnet", to jej użytkownicy będą mogli nie tylko dowolnie komunikować się ze sobą, ale także korzystać z usług pozostałych sieci tego typu. CSNET jest więc podobna w swojej uniwersalności do sieci telefonicznej — pozwala ludziom porozumiewać się poprzez komputer.

Sieć sieci

CSNET jest w istocie logiczną siecią łączącą kilka sieci fizycznych. Najbardziej podstawowa, to sieć PHONENET, która za pomocą dwóch głównych komputerów (Uniwersytet Delaware na Wschodnim Wybrzeżu i Rand Corp. na Zachodnim) działa jako elektroniczna poczta. Pozwala ona użytkownikowi wysłać wiadomość niezależnie od zajętości linii — w odpowiednim momencie komputer wywoła adresata i dostarczy „list". Tam z kolei, gdzie jest opłacalne stosowanie prawdziwej komunikacji między komputerami — są one przyłączone do sieci komutacji pakietów TELENET.

Oczywiście przyłączenie do tej sieci sprawia, że z każdego komputera można korzystać jak z własnego.

CSNET zapewnia ponadto przejście do ARPANET, łącząc wszystkich użytkowników obu sieci. Już samo to udogodnienie jest wystarczająco atrakcyjne dla tych badaczy, którzy chcieliby utrzymywać kontakt z kolegami, a nie pracują dla Pentagonu. Użytkownicy przyłączeni do PHONENET lub TELENET będą oczywiście mogli korzystać tylko z wymiany elektronicznej poczty z ARPANET, bez możliwości zdalnego uruchamiania programów i przesyłania zbiorów. Wyjątek stanowi 17 komputerów będących jednocześnie komputerami głównymi z ARPANET i w CSNET.

Sprawne funkcjonowanie elektronicznej poczty wymaga maksymalnych ułatwień w adresowaniu. Na uniwersytecie w Wisconsin opracowano system, który pozwala wysyłać wiadomości nie znając sieciowego adresu odbiorcy. Jest to po prostu baza danych obejmująca użytkowników sieci. Użytkownicy mogą sami uzyskiwać z tej bazy adresy, lecz nie jest to konieczne — system potrafi dokonać tego sam, już podczas operacji przesyłania.

Jak powstała CSNET

Pomysł sieci tego typu narodził się kilka lat temu. Profesor Lawrence Landweber z Wisconsin doszedł do wniosku, że społeczność informatyków, mała i rozproszona, potrzebuje czegoś bardzo podobnego do ARPANET.

„Mamy nadzieję, że takie szerokie możliwości komunikacyjne pozwolą geograficznie oddalonym od siebie ludziom pracować razem nad ≫ rozproszonymi projektami ≪" — stwierdził profesor. Mogą to być zarówno dyskusje nad matematycznymi podstawami informatyki, jak i rozwój praktycznych programów.

Fizyczne połączenia potrzebne do zrealizowania CSNET już istniały: ARPANET, GTE TELENET (publiczna sieć komutacji pakietów) oraz konwencjonalny system telefoniczny. Brakowało tylko oprogramowania, które pozwoliłoby komputerom używać wymiennie tych różnych sieci. A ponieważ komputery VAX (DEC) i system operacyjny UNIX rozpowszechniły się w środowiskach uniwersyteckich — one stały się podstawą nowej sieci. Profesor David Farber z uniwersytetu Delaware stworzył oprogramowanie PHONENET, które rozszerza możliwości komunikacji pomiędzy systemami UNIX o automatyczne adresowanie i przesyłanie w sieci. Naukowcy z uniwersytetu Purdue napisali programy tworzące sprzęg (ang. interface) między standardem protokołów komunikacyjnych TCP/IP (Departament Obrony) i protokołami X.25. Dodatkowo w Wisconsin powstają procedury umożliwiające podłączenie maszyn IBM System 370 i 4300 używających własnej wersji X.25. Natomiast oprogramowanie sieci PHONENET zostało w całości napisane w języku PASCAL, którego kompilatory są stosowane w wielu komputerach. Tak więc posiadając nawet najtańszy model PDP-11 (20 tys. $) wyposażony w system UNIX można stać się uczestnikiem CSNET. Przy umiarkowanym korzystaniu z tej sieci roczna opłata za użytkowanie wyniesie 20—25 tys. $.

Funkcjonowanie CSNET

Na co dzień funkcjonowaniem sieci zajmuje się Centrum Koordynacyjne i Informacyjne w firmie BOLT, BERANEK AND NEWMAN INC (BBN), tej samej, która opracowała i eksploatuje ARPANET. Natomiast nad całością polityki CSNET czuwa komitet złożony z prof. L. Landwebera i D. Farbera, przedstawicieli firm Rand Corp. i BBN oraz Narodowej Fundacji Naukowej — NSF. Nie wyklucza się też możliwości powołania niekomercyjnej organizacji, która będzie zarządzać siecią.

Prace nad uruchomieniem CSNET zostały uproszczone dzięki rezygnacji z jakichkolwiek mechanizmów ochrony — wszystkie zasoby są dostępne dla wszystkich. Istnieją przy tym obawy przed zasypywaniem użytkowników stosami niepotrzebnej, wytwarzanej przez komputery korespondencji, czyli przed zjawiskiem, które najbardziej irytuje uczestników ARPANET. Jednak profesor Landweber wierzy, że dzięki CSNET będzie istniało łatwe połączenie między dowolnymi informatykami na terenie całego kraju i że pierwsza sieć łącząca wszystkich ludzi z tej branży jest wielkim krokiem naprzód.

Oprac. MkS
na podstawie ELECTRONICS
z 20.10.1982

Słownictwo z zakresu inżynierii oprogramowania

W ciągu ostatnich kilkunastu miesięcy, oprócz terminologii języka ADA i terminologii odpowiadającej treści numerów monograficznych (system CAMAC, technika mikroprocesorowa), zajmowaliśmy się także terminologią dotyczącą oprogramowania (O oprogramowaniu podstawowym, nr 7/1982, O testowaniu, weryfikacji i uwiarygodnieniu programów, nr 8-9/1982). Ponieważ w ubiegłym roku opublikowano normę IEEE Std 729—1982, pn. „IEEE Standard Glossary of Software Engineering Terminology", co stanowi dość ważne wydarzenie w dziedzinie terminologii komputerowej, sądzę, że kontynuując tę tematykę warto uzupełniać i korygować odpowiednie terminy polskie oraz ich definicje.

W pierwszym artykule poruszającym te zagadnienia zajmiemy się terminami ściśle związanymi z powstaniem i rozwojem języka ADA, choć zakres ich stosowalności nie ogranicza się tylko do tego języka.

Według założeń narzuconych przez kontrahenta projektantom języka, jest on przeznaczony do programowania tzw. wbudowanych systemów komputerowych (ang. embedded computer systems). Choć nietrudno domyślać się, co to określenie oznacza, nie istniała dotychczas jego jednoznaczna definicja. W normie IEEE Std 729 stwierdza się, że wbudowany system komputerowy jest systemem komputerowym będącym częścią większego systemu, którego podstawowym celem nie są obliczenia („a computer system that is integral to a larger system, whose primary purpose is not computational").

Taka definicja, choć już przyjęta, jest jednak bardzo nieprecyzyjna, bowiem różnicuje systemy komputerowe na podstawie celu, któremu służą. Zwróćmy uwagę, że cel działania nie może być podstawą rozróżnienia, bowiem dokładnie taki sam system mógłby być wbudowany lub nie wbudowany, zależnie od celu, któremu służy. Należałoby zatem rozważyć strukturę systemów wbudowanych.

Dla dokładniejszego wyjaśnienia, co rozumie się przez wbudowany system komputerowy, podamy bardziej szczegółowe określenie systemu komputerowego w ogólności. Według wymienionej normy — jest to jednostka funkcjonalna, złożona z jednego lub wielu komputerów i odpowiedniego oprogramowania, używająca dzielonej pamięci dla całego programu lub jego części, a także dla całości lub części danych niezbędnych do wykonania programu. Wynika stąd, że wbudowane systemy komputerowe mogą być jedno- lub wieloprocesorowe, tzn. z dzieloną pamięcią, ale w zasadzie nie są to systemy wielokomputerowe, tzn. pozbawione dzielonej pamięci.

Takie rozróżnienie jest ostre, ale zastanówmy się, czy prawdziwe. Byłoby takie, gdybyśmy systemów bez wspólnej pamięci, tzn. wielokomputerowych, nie nazywali komputerowymi, co jest jawną sprzecznością zawartą zresztą w cytowanej definicji. Jeżeli natomiast dopuścimy rozłożenie funkcji systemu komputerowego na wiele jednostek obliczeniowych — tj. zarówno procesorów, jak i komputerów — to rodzi się pytanie, jaki system komputerowy nie będzie wbudowany.

Reasumując, żaden z obu elementów składowych definicji wbudowanego systemu komputerowego, tzn. ani stwierdzenia o celu działania, ani zawarte implicite stwierdzenie dotyczące współużywalności pamięci, nie różnicuje go względem innych systemów komputerowych.

Jak z tego wybrnąć? Wydaje się, że jedynym sposobem na sprecyzowanie definicji jest wyróżnienie innej istotnej cechy systemu komputerowego, świadczącej o fakcie wbudowania go w inny większy system. Tą cechą mogłoby być skupienie lub rozprzestrzenienie jednostek obliczeniowych. W takim przypadku można by lepiej odróżnić systemy wbudowane od tzw. rozłożonych przestrzennie (ang. distributed systems), stanowiących jakby naturalne przeciwieństwo systemów wbudowanych.

Omawianemu terminowi poświęciliśmy wiele miejsca — jako ściśle związanemu z językiem ADA. W żadnej mierze nie jest to jednak termin podstawowy dla tego języka, co zresztą rozumie się samo przez się — odnosi się on przecież do sprzętu. Spośród terminów związanych z oprogramowaniem warto wyróżnić kilka, dotyczących bardzo ważnej cechy języka, jaką jest operowanie danymi.

Podstawowe dla wszystkich języków programowania pojęcie typu danych zdefiniowano następująco: typ danych jest to klasa danych określona przez zbiór elementów i operacji, które mogą być na nich wykonywane. Ponieważ języki mogą istotnie różnić się pod względem roli, jaką odgrywają w nich typy danych, wprowadzono pojęcie stopnia typizacji języka, a dokładniej — zdefiniowano tzw. silną typizację (ang. strong typing). Według wymienionej normy, silna typizacja jest właściwością języka programowania, polegającą na konieczności deklarowania typów każdego obiektu (danych) i wykluczającą użycie operatora do niewłaściwego obiektu, wskutek czego zapobiega się wzajemnemu oddziaływaniu obiektów niezgodnych typów.

Oba wymienione pojęcia wiążą się z ważną cechą współczesnych języków programowania, jaką jest tzw. abstrahowanie (tzn. wyodrębnianie) danych, ang. data abstraction, które polega na wydzieleniu i zachowaniu jedynie podstawowych, charakterystycznych właściwości danych przez zdefiniowanie poszczególnych typów danych i odpowiadających im cech funkcjonalnych, dzięki czemu uzyskuje się oddzielenie i przesłonięcie szczegółów reprezentacji (wg definicji zawartej w omawianej normie). Przesłanianie informacji (ang. information hiding) jest techniką „kapsułkowania" (ang. encapsulation) decyzji projektowych w programowaniu za pomocą modułów w taki sposób, że sprzęgi modułów (ang. interface) zawierają możliwie najmniej szczegółów dotyczących ich wewnętrznego działania. Kapsułkowanie z kolei, jest techniką polegającą na wyodrębnieniu funkcji systemu wewnątrz modułu i podaniu dokładnej specyfikacji modułu.

Jedną z konstrukcji języka, umożliwiającą abstrahowanie danych, przesłanianie informacji i skuteczne kapsułkowanie, są tzw. pnie (ang. stub). Według omawianej normy, jedno ze znaczeń terminu pień, odpowiadające językowi ADA, jest następujące: instrukcja programu zastępująca ciało (ang. body) jednostki programowej i wskazująca, że jednostka jest lub zostanie zdefiniowana gdzie indziej.

Spośród innych terminów ściśle związanych z językiem ADA, lecz bezpośrednio nie dotyczących operowania danymi, zdefiniowano dwa: exception i rendezvous.

Choć nie ma zgodności co do tego, jak tłumaczyć na język polski słowo exception (por. W. Iszkowski, Terminologia języka ADA — nieco inaczej, INFORMATYKA, nr 3/1983), sądzę, że najmniej kontrowersyjne, możliwe do przyjęcia przez wszystkich, jest naturalne znaczenie tego słowa, tzn. wyjątek. Tak więc, wg wymienionej normy, wyjątkiem nazywa się zdarzenie, które powoduje zawieszenie normalnego wykonywania programu („an event that causes suspension of normal program execution").

O ile termin exception występuje prawie we wszystkich znanych językach programowania, to ostatni z omawianych terminów, rendezvous, jest specyficzny dla języka ADA. Sądzę, że nie popełnimy błędu tłumacząc to słowo na język polski jako spotkanie, choć wysuwa się też propozycje, aby go nie tłumaczyć i zachować brzmienie oryginalne. Spotkanie jest to oddziaływanie zachodzące między dwoma równoległymi zadaniami, gdy po wywołaniu wejścia jednego zadania przez drugie, zadanie wywołane wykonuje odpowiednią instrukcję przyjęcia (ang. accept) w imieniu zadania wywołującego.

JANUSZ ZALEWSKI

IBM PERSONAL COMPUTER przynosi korzyści nie tylko producentowi. Kilkunastomiesięczna obecność na rynku zaowocowała wieloma produktami pochodzącymi z niezależnych firm. Są to np.:
— usprawniony program redagowania ekranu
— system zarządzania bazą danych, zawierającą dodatkowo mechanizmy dostępu do danych w językach FORTRAN i PASCAL, GARDEN — specjalny język do manipulowania danymi, MARVEST — język zapytań oraz BLOOM — do formowania globalnych raportów z bazy
— kompilator języka COBOL wraz z interakcyjnym programem usuwania błędów.

Oprócz tego, umieszczenie specjalnych otworów w obudowie, które producent przewidywał do ewentualnego rozszerzania pamięci operacyjnej, wywołało lawinę różnych rozwiązań, np.:
— zestaw płytek z jednostkami centralnymi innymi niż standardowa, pozwalającymi korzystać z innych systemów operacyjnych, tzn. Z80 i CP/M, 68000 i XENIX 3.0, lub usprawniającymi działanie (INTEL 8086 i 8087)
— przystosowanie do współpracy z siecią telefoniczną — syntezer mowy oraz automat wybierający numery
— generator znaków alfabetu chińskiego; pozwala tworzyć do 24 tys. znaków
— rozszerzone funkcje wejść-wyjść analogowych i cyfrowych, przystosowujące komputer do pracy jako system zbierania danych i sterowania procesami.

Firma COLBY COMPUTER CORP. oferuje natomiast walizeczkę, która zawiera monitor ekranowy (przekątna ok. 20 cm), połączenia, zasilacz i miejsce na jednostkę dyskową. Pozwala ona, po wyjęciu z IBM PC trzech płytek i włączeniu ich do zestawu, otrzymać komputer przenośny.

*

Wprowadzając w tym roku na rynek nowe modele komputerów, firma APPLE COMPUTER INC. musi spodziewać się twardej walki o klientów. Walka tego typu nie jest niczym nowym w tak burzliwie rozwijającej się dziedzinie mikrokomputerów, jednak w tym przypadku zapowiada się konkurencja między wyrobami tego samego producenta. Na początku roku, oprócz zmodyfikowanych wersji APPLE II PLUS i APPLE III (w cenie odpowiednio 1500 i 4000 $), rozpoczęto sprzedaż całkowicie nowego modelu LISA (8—10 tys. $). LISA, zbudowana w oparciu o 16-bitowy procesor 68000, to wysokiej klasy komputer do zastosowań biurowych: 512 K do 1 M bajta pamięci operacyjnej, dwie stacje minidyskietek po 870 K bajtów, ekran o wysokiej rozdzielczości oraz „myszka". W cenie mieści się również oprogramowanie: pakiety do przetwarzania tekstów, obliczeń, graficzne, komunikacyjne, do zarządzania projektami i system bazy danych. Najważniejszą cechą tych programów jest możliwość wzajemnego przekazywania sobie informacji — uzyskano to dzięki temu, że każdy z nich posługuje się tym samym graficznym sposobem reprezentacji danych. Na trzeci kwartał zapowiedziano premierę jeszcze jednego modelu: braciszka LISY o nazwie MACINTOSH. Podobnie, choć nie tak bogato wyposażony, ma zachować zgodność oprogramowania z LISĄ. Przewidywana cena — 2000 $ — może właśnie spowodować, że stanie się najgroźniejszym rywalem LISY.

*

Niemałe zaskoczenie wywołał nowy model komputera osobistego firmy TEXAS INSTRUMENTS CORP., gdyż ten znany producent układów scalonych zastosował w nim mikroprocesor INTEL 8088. PEGASUS, bo tak go nazwano, jest przeznaczony do zastosowań profesjonalnych, a decyzja wybrania układu pochodzącego z konkurencyjnej firmy została podyktowana chęcią wykorzystania oprogramowania stworzonego dla IBM Personal Computer. Jednocześnie jednak TEXAS INSTRUMENTS rozpoczyna wytwarzanie dwóch komputerów w oparciu o własne mikroprocesory. Pierwszy z nich, CC-40, to wyrób z dopiero tworzącej się klasy komputerów przenośnych (waga ok. 0,5 kg). Urządzenie to, zasilane czterema bateriami typu R6, ma 31-znakowy wyświetlacz na ciekłych kryształach, 32 K bajty pamięci ROM (interpreter języka BASIC), 6 K bajtów (z możliwością rozszerzenia do 16 K) adresowanej przez użytkownika pamięci RAM oraz procesor 8-bitowy. Rozpoczęcie produkcji CC-40 wiąże się z zaniechaniem planów wprowadzenia na rynek nowego kalkulatora programowego. Cena, ustalona w wysokości ok. 250 $, jest niska, jednak prawdziwym szokiem ma być model 99/2 — pierwszy 16-bitowy komputer przełamujący barierę 100 $ (dokładnie 99,95 $). Oczywiście 99/2 to urządzenie przeznaczone dla początkujących, a zwłaszcza do użytku domowego — w wersji podstawowej ma tylko 4,2 K bajta RAM. Przewiduje się jednak, że można będzie je wzbogacać np. o drukarkę, jednostkę taśmy magnetycznej modem czy dodatkowe moduły pamięci operacyjnej (do 36,2 K).

*

Koszty prac badawczo-rozwojowych w dziedzinie półprzewodników i sprzętu informatycznego stały się zbyt wysokie nawet dla największych firm. Z inicjatywy Roberta M. Price'a, prezesa firmy CONTROL DATA, powołano konsorcjum Microelectronics and Computer Technology Corp. (MCC), które ma zajmować się wielkimi programami badawczymi na rzecz swoich udziałowców. Wśród założycieli MCC znalazły się tak znane firmy, jak: ADVANCED MICRO DEVICES INC., DIGITAL EQUIPMENT, HARRIS, NCR, RCA i SPERRY. Obecnie prowadzona jest akcja rekrutacji personelu, którego część zostanie zapewne oddelegowana z firm-założycielek.
Zapowiedziano już cztery pierwsze programy badawcze: architektura komputerów, projektowanie i produkcja wspomagana komputerem (CAD/CAM), metody wytwarzania i łączenia systemów i podzespołów oraz wydajność oprogramowania. Ponieważ nie wybrano jeszcze siedziby dla MCC, można się spodziewać, że oprócz kwatery głównej firma będzie mieć oddziały prowadzące poszczególne tematy, a zlokalizowane w miejscach koncentracji odpowiedniego przemysłu i kadry badawczej. Każde z dotychczas określonych przedsięwzięć obliczone jest na 5 do 10 lat, natomiast budżet MCC wyniesie 50—100 mln $. Ponieważ już sam projekt takiego konsorcjum wywołał obawy, czy nie zachodzi tu przypadek łamania ustawy antytrustowej, B.M. Price wezwał Kongres St. Zjednoczonych do uproszczenia jej przepisów w dziedzinie wspólnych organizacji badawczo-rozwojowych. W przeciwnym przypadku trudno będzie zachować konkurencyjność amerykańskiego przemysłu.

*

Firma DIGITAL RESEARCH INC., sprzedawca popularnego systemu operacyjnego CP/M, zapowiedziała wprowadzenie do 16-bitowych wersji własnego kompilatora języka C. Pozwoli to na przykład uruchamiać pod nadzorem CP/M programy napisane na maszynach wyposażonych w system UNIX. Kompilator C będzie dostępny dla CP/M 68K (procesor 68000) oraz CP/M 86 (procesor INTEL 8086).

*

Współbieżna sieć układów Transputer firmy INMOS (INFORMATYKA 5/83) utworzy jednostkę centralną 32-bitowego komputera biurkowego konstruowanego w Britain's Imperial College. Maszyna ta, nazwana komputerem do języków aplikacyjnych, prowadzi obliczenia w językach piątej generacji (np. w PROLOGU) z szybkością o rząd wielkości większą niż nowoczesny obecnie minikomputer VAX (DEC). Uzyskano to dzięki temu, że architektura została dobrana specjalnie pod kątem języków zawierających współbieżność, dzięki czemu instrukcje wykonuje się równolegle, a nie sekwencyjnie, jak w klasycznej maszynie von Neumanna. Prototyp ma 12 M bajtów pamięci podzielonej na 16 segmentów i obsługującej 16 procesorów, z których każdy jest zbudowany z trzech układów TRANSPUTER. Półprzewodnikowa sieć może łączyć każdy procesor z każdym segmentem pamięci.

*

Już w tym roku IBM zamierza rozszerzyć gamę produkowanych komputerów osobistych. W lecie oczekuje się premiery „rodzinnego" komputera PEANUT (w cenie 600—750 $). Następnie powinien pojawić się na rynku PERSONAL COMPUTER 2 wyposażony w wielobarwny ekran o wysokiej rozdzielczości. Inna nowość, to profesjonalny ROVER (pamięć operacyjna do 1 M bajta, dysk Winchester 5 M bajtów), kompatybilny z terminalem konwersacyjnym IBM 3270. Będzie on posiadał płaski ekran plazmowy o pojemności 2 tys. znaków. Ostatnia z zapowiedzi, to wieloprogramowy super-komputer osobisty POPCORN, który ma mieścić się w tej samej klasie co LISA firmy APPLE. Ale LISA jest już w sklepach, a POPCORN w pracowniach konstruktorów.

Huzar A.: Programowanie współbieżne w systemie OS/JS na poziomie PASCALA

INFORMATYKA 1983, nr 6, s. 2

Dostosowanie języka PASCAL do rozwiązywania problemów programowania współbieżnego na komputerze R-32. Omówiono sposób uzyskania współbieżności oraz podano przykład programu współbieżnego i zasady przydziału zasobów.

Хузар А.: Параллельное программирование в системе ОС ЕС на уровне языка PASCAL

INFORMATYKA 1983, № 6, стр. 2

Приспособление языка PASCAL к решению проблем параллельного программирования на ЭВМ R-32. Обсуждён способ получения параллельности, а также приведен пример параллельной программы и правила распределения ресурсов.

Owczarczyk J.: Projektowanie systemu graficznego. Część 1

INFORMATYKA 1983, nr 6, s. 5

Propozycja zaprojektowania i realizacji rastrowego systemu graficznego metodą rozszerzenia konfiguracji posiadanego sprzętu komputerowego. Omówiono założenia wstępne, procesor oraz pamięć obrazową.

Овчарчык Й.: Проектирование графической системы. Часть 1

INFORMATYKA 1983, № 6, стр. 5

Предложение запроектирования и реализации растровой графической системы методом расширения конфигурации имеющегося аппаратного оборудования вычислительной системы. Обсуждены вступительные предположения, процессор и образная память.

Grabowski J.: Przegląd mikroprocesorów 16 bitowych (4). TEXAS INSTRUMENTS 9900

INFORMATYKA 1983, nr 6, s. 8

Czwarta część charakterystyki najbardziej popularnych mikroprocesorów 16-bitowych, obejmująca omówienie mikroprocesora TEXAS INSTRUMENTS 9900. Scharakteryzowano rejestry, tryby adresowania, listę rozkazów oraz przerwania programu.

Грабовски Й.: Обзор 16-битовых микропроцессоров (4).

INFORMATYKA 1983, № 6, стр. 8

Четвёртая часть характеристики найболее популярных 16-битовых микропроцессоров содержащая обсуждение микропроцессора TEXAS INSTRUMENTS 9900. Охарактеризованы регистры, режимы адресации, набор команд и прерывания программы.

Iszkowski W., Maniecki M.: Standaryzacja języka BASIC (2). Moduł rdzeniowy języka

INFORMATYKA 1983, nr 6, s. 16

Projekt międzynarodowej normy pełnego języka BASIC. Scharakteryzowano poszczególne elementy modułu rdzeniowego tej wersji języka: tekst programu, operacje i funkcje numeryczne, instrukcje sterujące i iteracyjne, operacje napisowe, modularność programu, system plików, instrukcje wprowadzania i wyprowadzania, instrukcje tablicowe oraz obsługę protestów.

Ишковски В., Манецки М.: Стандартизация языка BASIC (2). Ядро языка

INFORMATYKA 1983, № 6, стр. 16

Проект международного стандарта полного языка BASIC. Охарактеризованы отдельные элементы ядра этой версии языка: текст программы, числовые операции и функции, инструкции управления и повторения, текстовые операции, модульность программы, система файлов, инструкции ввода/вывода, инструкции обработки таблиц и обслуживание исключений.

Oktaba H.: Prefiksowanie klasami w LOGLANIE

INFORMATYKA 1983, nr 6, s. 20

Charakterystyka koncepcji prefiksowania klasami jako mechanizmu modularyzacji programów w języku LOGLAN. Podano definicję pojęcia oraz omówiono sposoby i zalety prefiksowania ilustrując je prostymi przykładami.

Октаба Г.: Префиксация классами на языке LOGLAN

INFORMATYKA 1983, № 6, стр. 20

Характеристика конценции префиксации классами как механизма модуляризации программ на языке LOGLAN. Даётся определение понятия и обсуждаются способы и достоинства префиксации иллюстрированные простыми примерами.

Postół M., Staniaszczyk J., Weber W., Zdanowski A.: Sprzęgi systemowe w urządzeniach mikrokomputerowych

INFORMATYKA 1983, No. 6 p. 24

Analiza trudności związanych z opracowaniem uniwersalnego sprzęgu dla mikrokomputerów. Podano kryteria wyboru sprzęgu oraz wskazano na znormalizowane już sprzęgi międzynarodowe odpowiadające tym kryteriom, ilustrując przykładem rozwiązania sprzęgu MULTIBUS.

Постул М., Станящык Й., Вебер В., Здановски А.: Системные интерфейсы в микрокомпьютерных устройствах

INFORMATYKA 1983, № 6, стр. 24

Анализ трудностей связанных с разработкой универсального интерфейса микро-ЭВМ. Даны критерии выбора интерфейса и указаны уже стандартные международные интерфейса отвечающие этим критериям, с иллюстрирующим примером решения интерфейса MULTIBUS.

Huzar A.: Concurrent programming in the OS/JS operating system at PASCAL level

INFORMATYKA 1983, No. 6, p. 2

PASCAL adaptation for solving concurrent programming problems on R-32 computer. Methods for concurrency achievement as well as concurrent program example and rules for resource allocation are discussed.

Huzar A.: Parallelprogrammierung im OS/JS Betriebssystem auf PASCAL-Niveau

INFORMATYKA 1983, Nr. 6, S. 2

Anpassung der PASCAL-Sprache zur Lösung der Parallelprogrammierung auf R-32 Rechner. Es wurden die Erreichungsmethoden der Parallelarbeit, sowie ein Beispiel des Parallelprogrammes und Regeln für die Resourceszuteilung besprochen.

Owczarczyk J.: Graphic computer system design. Part 1

INFORMATYKA 1983, No. 6, p. 5

Proposal for design and realization of graphic computer system with raster organization based on extension of existing hardware configuration. Preliminary assumptions, processor and image store are discussed.

Owczarczyk J.: Projektierung des graphischen Rechnersystems. Teil 1

INFORMATYKA 1983, Nr. 6, S. 5

Ein Vorschlag für Projektierung und Bau eines graphischen Rechnersystems mit Rasterorganisation mittels Erweiterung der bestehenden Rechnerkonfiguration. Es wurden Voraussetzungen, Prozessor und Abbildungsspeicher besprochen.

Grabowski J.: 16-bit microprocessor survey (4). TEXAS INSTRUMENTS 9900

INFORMATYKA 1983, No. 6, p. 8

Fourth part of the most popular 16-bit microprocessor characteristics — presentation of TEXAS INSTRUMENTS 9900 microprocessor. Registers, addressing modes, instruction set and program interrupts are characterized.

Grabowski J.: Vorschau von 16-Bits-Mikroprozessoren (4). TEXAS INSTRUMENTS 9900

INFORMATYKA 1983, Nr. 6, S. 8

Vierter Teil der Charakteristik der populärsten Vertreter von 16-Bits-Mikroprozessoren, die eine Besprechung von TEXAS INSTRUMENTS 9900 umfasst. Es wurden Register, Adressierungsmodi, Befehlsliste und Programmunterbrechungen besprochen.

Iszkowski W., Maniecki M.: BASIC language standarization (2). Core module of the language

INFORMATYKA 1983, No. 6, p. 16

International standard proposal for complete BASIC. Individual components of the core module for this language version are characterized: program's text, numeric operations and functions, control and iterative statements, string operations, program's modularity, file system, I/O statements, table statements and exeption handling.

Iszkowski W., Maniecki M.: BASIC-Sprache Normierung (2). Kernmodul der Sprache

INFORMATYKA 1983, Nr. 6, S. 16

Eine Charakteristik des internationalen Normentwurfes für volle Version der BASIC-Sprache. Es wurden die einzelnen Elemente des Kernmodules von dieser Sprachevarsion charakterisiert: Text des Programmes, numerische Operationen und Funktionen, Steuer- und Iterationsbefehle, Aufschriftsoperationen, Programmodularheit, Dateisystem, E/A- und Tafelbefehle sowie Ausnahmenhandhabung.

Oktaba H.: Classes prefixation in LOGLAN

INFORMATYKA 1983, No. 6, p. 20

Characteristics of classes prefixation idea as mechanism for program's modularization in LOGLAN language. Definition of the term as well as methods and advantages of prefixation, illustrated with simple examples, are discussed.

Oktaba H.: Klassenprefixierung in LOGLAN

INFORMATYKA 1983, Nr. 6, S. 20

Eine Charakteristik der Klassenprefixierungkonzeption als Mechanismus für Modularisierung der Programme in LOGLAN-Sprache. Es wurden Definition des Begriffes angegeben, sowie Methoden und Vorteile der Prefixierung, illustriert mit einfachen Beispielen, besprochen.

Postół M., Staniaszczyk J., Weber W., Zdanowski A.: System interfaces in microcomputer devices

INFORMATYKA 1983, No. 6 p. 24

Analysis of difficulties connected with elaboration of microcomputer universal interface. Selection criteria are presented, and standarized international interfaces fulfilling these criteria are illustrated with an example of MULTIBUS interface solution.

Postół M., Staniaszczyk J., Weber W., Zdanowski A.: Systemschnittstellen in Microrechnereinrichtungen

INFORMATYKA 1983, Nr. 6 S. 24

Eine Analyse von Schwierigkeiten, die mit Erarbeitung einer universellen Schnittstelle für Mikrorechner verbunden sind. Es wurden Schnittstelleauswahlkriterien angegeben und die entsprechend dieser Kriterien schon international genormten Schnittstellen mit einem Lösungsbeispiel von MULTIBUS-Schnittstelle, angezeigt.

Dylemat „skłóconych"

POLITYKA opublikowała niedawno (nr 24 z 11.06.83) fragment książki Pawła Bożyka. Frapujący fragment o informatyce, informatykach i ich wkładzie w kryzys gospodarczy Polski.

Jakże tu dyskutować z autorem, który mając dostęp do różnych tajnych dokumentów (np. do materiałów Komisji Partyjno-Rządowej ds. Informatyki) — zaczerpnął skądś sumę 120 mld zł przeznaczonych na rozwój informatyki w latach 1971—1980. W rzeczywistości zaś (biuletyn GUS) okazuje się, że w roku 1981 cały nasz sprzęt informatyczny wart był jedynie ok. 35 mld, przy czym wlicza się w to niektóre eksploatowane jeszcze komputery z lat sześćdziesiątych. Gdyby dodać jeszcze 12 mld zł obiecane przemysłowi komputerowemu na rozwój potencjału produkcyjnego, a obcinane w kolejnych latach zadyszki — do ok. 6 mld, to i tak ostateczne wielkości nie osiągają łącznie nawet 50% wspomnianej kwoty.

O innych sprawach poruszonych przez Pana Profesora równie trudno jest dyskutować. Błędy i przeinaczenia [1]) nie są zbyt wielkie (ot tak pół na pół z prawdą), jest za to atmosferka wywodów oraz ciekawostki — niby takie sobie, ale zawsze uroczo dopełniające szkicowany portret.

Sądzę, że dalej warto się zająć jedynie przyjętym przez Pawła Bożyka, zasłyszanym w kuluarach jeszcze w czasach „świetności", poglądem o środowisku informatyków — „należącym do najbardziej skłóconych w kraju".

Skąd to „skłócenie"? Zacznijmy od początku, od początku dekady przyspieszonego rozwoju, łatwych (ale nie dla wszystkich) dolarów. Informatyka w Polsce startowała istotnie z dość niskiego poziomu, niemniej hasła i rzeczywistość pierwszych lat dekady owocowały i w naszej informatyce totalnymi wizjami, najpierw powszechnej informatyzacji (ponętnej nad wyraz dla centralnych decydentów), a później powszechnej (optymalnej oczywiście) automatyzacji zarządzania.

Nie byłoby niczego dziwnego w tych wizjach, gdyby nie fakt, że zostały tak skwapliwie podchwycone przez bardzo wysokie persony i że tak ochoczo zabrano się do dzieła.

A co do tego ma środowisko? Ano właśnie nic. Nie miał znaczenia fakt, że wypowiada się ono z sarkazmem o wizjach, że zabiera się za ich realizację bez odrobiny wiary w realność poczynań. Na kilku zorganizowanych naradach (znów nie ze środowiskiem) ostro rozprawiono się z nieśmiałymi głosami wątpiących, zarzucając im małoduszność, brak umiejętności, niedwuznacznie zapowiadając sankcje personalne wobec pozostających w tyle. Zresztą, z jakimż to wówczas środowiskiem dyskutowano lub jakie pytano o zdanie, wystarczyły do konsultacji mianowane rady, mianowani dyrektorzy, mianowani fachowcy.

Co nieco przenikało jednak do uszu ważnych person, bywały bowiem i spektakularne sprzeciwy wobec totalnych perspektyw. Wówczas najprościej było posłużyć się niby obok rzuconym wyjaśnieniem, że to tylko kłótliwość i niedojrzałość, a także wątpliwe morale środowiska — przeszkadzają w szybkim marszu do zinformatyzowanego (i być może optymalnie zautomatyzowanego) zarządzania państwem. Takich sformułowań Pan Profesor nie użył, ale czymże są jego wywody o swarliwości informatyków pchających się do wyjazdów zagranicznych. Iluż w końcu z nich wyjechało? Niewiele pomylę się twierdząc, że 50% wyjeżdżających na szkolenie to nie informatycy, a — nominaci, kibice lub po prostu wpływowe osóbki.

Usprawiedliwieniem wizjonerów mogło być to, że i w bliskim nam kraju, również pojawiały się głosy o takich możliwych perspektywach. Co jednak może usprawiedliwiać decydentów? Niestety, niczego sensownego na usprawiedliwienie nie znajduję. Marny to bowiem decydent, jeśli wystarczają mu choćby i natchnione wizje czynione przez natchnione grono, a nie wysłuchuje sceptyków i nie zleca biegłym w rachunkach, koniunkturach, rynkach i cenach sprawdzenia podstaw podejmowanej decyzji.

Rozgoryczenie decydentów — po dość szybkim przebudzeniu — nie miało tylko skutków werbalnych; miało skutki konkretne i to jeszcze dziś niezmiennie trwające. Opinia o informatyce „obnażającej socjalizm" — wygłoszona w formie złowieszczego żartu przez Premiera na ostatnim posiedzeniu Komitetu Informatyki w 1977 r. utwierdziła mniejsze i większe osobistości, że coś z informatykami jest nie tak, nie mogą być przecież winne te tak genialne komputery.

W ostatnich latach, opinia o podejrzanych informatykach ponownie tu i ówdzie odżywała — a to że ekstremiści, a to że siedlisko niepewnych inicjatyw itd.

O cóż to informatykom chodziło i chodzi? Po pierwsze (i przede wszystkim) — o fachowość; ona to bowiem uczyni informatykę i informatyków przydatnymi, tak jak to się dzieje na całym świecie. Fachowe poglądy i kierunki rozwoju nie mogą się rodzić w gabinetach i w formie jednostronnych wizji.

Jeszcze w 1968 roku rozpoczęte zostały starania o powołanie stowarzyszenia zawodowych informatyków, o stworzenie środowiska fachowców, zdolnego wypowiadać sądy wyważone i oparte na wiedzy. Starania te dopiero w 1981 roku zakończyły się powodzeniem — powstało Polskie Towarzystwo Informatyczne.

I tu zaskoczenie. Memoriał opracowany przez PTI wspólnie z Komitetem Informatyki PAN (nie należy go mylić z martwym de facto Komitetem, na którego czele w przeszłości stał premier) mówił to samo, co Paweł Bożyk w końcowym akapicie artykułu wydrukowanego w POLITYCE. Memoriał rozesłany dość dawno do wielu najwyżej postawionych osób oraz instytucji wskazywał na konieczność zajęcia się informatyką przez władze państwowe. Nie było tam mowy o tzw. koordynacji, ale o konieczności kształtowania przez Państwo rozwoju informatyki i jej zastosowań. Wskazywano, że narzędzia ekonomiczne oferowane przez reformę gospodarczą świetnie nadają się nie tylko do eliminowania nierealnych przedsięwzięć, ale — jak wskazują ościenne, bliższe i dalsze przykłady — dają informatyce impulsy na tyle skuteczne, że staje się ona znaczącym (i pożądanym) elementem gospodarki i życia społeczeństw.

Jeszcze raz okazało się — i to w życiu, nie w matematyce — że z przesłanek niezupełnie poprawnych można niekiedy również, tak jak zawsze z przesłanek poprawnych, dojść do poprawnych wniosków.

Jeśli konkluzja Pawła Bożyka miałaby siłę przebicia większą niż wspomniany memoriał — to dobrze, znakomicie. Sojuszników, niestety, najczęściej się nie wybiera, niemniej każdy z nich może się stać pożyteczny.

JANUSZ GWIAZDA

[1]) Na przykład — komputery serii ODRA 1300 (i wszystkie inne o nazwie ODRA) nigdy nie były produkowane przez Instytut Maszyn Matematycznych

Zasady prenumeraty

Zamówienia i przedpłaty na prenumeratę INFORMATYKI przyjmuje Zakład Kolportażu Wydawnictwa NOT SIGMA. Adres pocztowy: Wydawnictwo NOT SIGMA — Zakład Kolportażu, 00-950 Warszawa, skr. poczt. 1004. Konto bankowe: 1036-7490-139-11, III O/M NBP w Warszawie.

Jednostki gospodarki uspołecznionej, instytucje i organizacje przesyłają zamówienia (w 1 egz.) zawierające: tytuł czasopisma, liczbę zamawianych egzemplarzy, okres prenumeraty i pełny adres zamawiającego z kodem pocztowym, a także numer konta bankowego zamawiającego oraz (ewentualnie adresy odbiorców, którzy na zlecenie zamawiającego mają prasę otrzymywać.

Dopisując w zamówieniu PRENUMERATA STAŁA, zamawiający nie będzie musiał corocznie ponawiać zamówienia, a jedynie dokonywać przedpłaty według aktualnie obowiązujących cen.

Warunkiem realizacji zamówienia jest równoczesne dokonanie odpowiedniej wpłaty na ww. konto Wydawnictwa NOT SIGMA.

Prenumeratorzy indywidualni dokonują wpłaty przekazem na ww. konto, podając na odwrocie odcinka dla adresata-posiadacza rachunku: tytuł czasopisma, liczbę zamawianych egzemplarzy oraz okres prenumeraty.

Zamówienia i wpłaty przyjmowane są w terminach:

- do 15 listopada — na I kwartał, I półrocze i cały rok
- do 28 lutego — na II kwartał
- do 31 maja — na III kwartał i II półrocze
- do 31 sierpnia — na IV kwartał.

Uwaga: przy podawaniu kodu pocztowego i numeru konta bankowego obowiązuje bardzo czytelne pismo.

Prenumerata kwartalna — 225 zł, półroczna — 450 zł, roczna — 900 zł. Prenumerata ze zleceniem wysyłki za granicę jest dwukrotnie droższa.

Od 1 lipca br. wprowadzona została prenumerata ulgowa dla członków stowarzyszeń naukowo-technicznych NOT, studentów wyższych uczelni, uczniów szkół zawodowych. Warunkiem jej uzyskania jest poświadczenie blankietu dla nabywcy indywidualnego (na odcinku dla adresata) przez właściwe stowarzyszenie NOT, wyższą uczelnię lub szkołę zawodową. Cena prenumeraty ulgowej: kwartalnie — 150 zł, półrocznie — 300 zł, rocznie — 600 zł.

Dodatkowych informacji o prenumeracie udziela: Zakład Kolportażu, tel. 40-00-21 w. 293, 295, 217 oraz 40-35-89. Egzemplarze archiwalne można nabywać w Klubie Prasy i Informacji Technicznej w Warszawie, ul. Mazowiecka 12, tel. 27-43-65.

W numerze 7-8/83 przedstawimy:

Raport dla Klubu Rzymskiego — Barnaby F.: Zastosowanie mikroelektroniki w urządzeniach wojskowych

Abramowicz A. W.: Automatyczne redagowanie tekstów

Liszyński Z.: Programowanie procesów współbieżnych. Środowisko sprzętowe

Grabowski J.: Przegląd mikroprocesorów 16-bitowych (5). LSI-11

Wierzbowski J. (oprac.): Wielkie wydarzenia w rozwoju oprogramowania

Fabijanek A., Marciniak K., Wojciechowski J.: GOBI — system do obliczeń geometrycznych

Fladrowska E.: Warunki i metody efektywnego zastosowania podsystemu TPP

Dziedzic M., Perycz K., Wiliński J.: Wieloprogramowy dyskowy symulator ODRY 1305 na R-32

Tadeusiewicz R., Pachowicz P.: CESARO — system analizy i rozpoznawania obrazów wizualnych

W najbliższych numerach:

- Sieci teleinformatyczne
- Język SIMBOL
- Programowanie za pomocą współprogramów
- Przemysłowy FORTRAN czasu rzeczywistego
- Programowane systemy pomiarowe
- Języki symulacyjne
- Jeszcze o ZX81